GENERAL INSTRUMENT	PIC7001 PIC7041

PRELIMINARY INFORMATION

8 BIT MICROCOMPUTER

FEATURES:

- Pin for pin compatible with the IMS7001/7041
- On-board serial port with internal/external baud rate generator
- 8-bit instruction data word
- Microprocessor and/or Emulator - PIC7001
- 4096 bytes of on-chip ROM - PIC7040
- External memory expansion capability
- 128 memory-mapped registers
- Full-feature data/program stack
- Two external maskable interrupts
- 2 On-chip 13 bit timer/event counter with interrupt and capture latch
- Separate baud rate timer usable as a third timer
- 32 bits general purpose input/output
- N-channel silicon gate MOS
- 5V power supply
- TTL compatible I/O pins
- Single instruction BCD add and subtract
- Microprogramable instruction set

DESCRIPTION

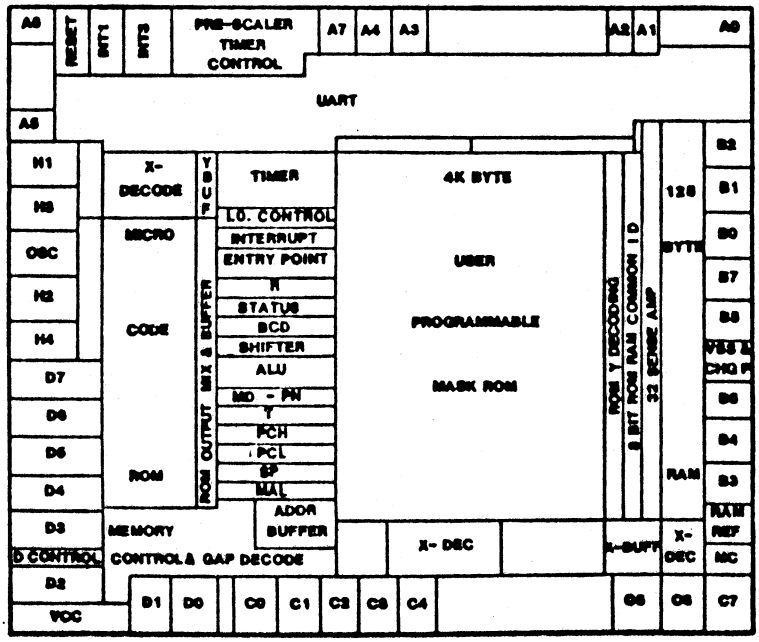
The PIC7041 is single-chip, a 8 bit microcomputer containing CPU, ROM, RAM, and I/O and serial port. The PIC7001 contains CPU, RAM, and I/O on-chip, serial port and functions as a very powerful microprocessor. The PIC7041 provides four kilobytes of on-chip ROM. Both devices have the capability, through memory expansion modes, to access up to 64 kilobytes of external address space.

Each device has 128 bytes of RAM, a standard 61 command instruction set, and both register and memory-based architecture. The PIC7000 series uses MOS memory cell structure to replace most of the random logic other microcomputers use to implement functions.

The density of these memory cells, and their ease of interconnection, make for a small but powerful microcomputer. The organization of the memory cell architecture causes the interconnect line to run in ordered patterns known as Strip Chip Architecture Topology. This also reduces the amount of chip area required for the interconnections.

The choice of five memory expansion modes allows the user to fit any member of the PIC7000 series to the requirements of a particular system. The 32 I/O pins may be used in the single chip configuration, or easily traded off for interfaces to peripheral devices or external memories. The user chooses the desired mode with two bits in the I/O control register and a single hardware connect.

PIC7041 SERIES BLOCK DIAGRAM



1984 General Instrument Corporation

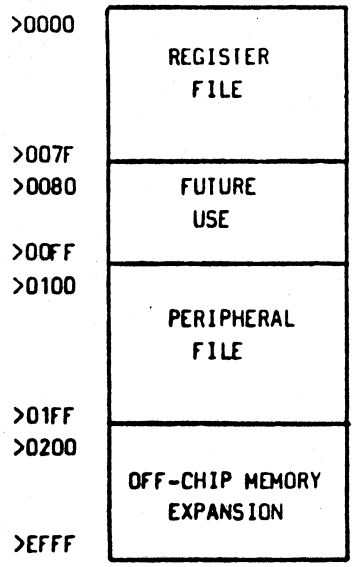
ARCHITECTURAL DESCRIPTION

Registers And Memory

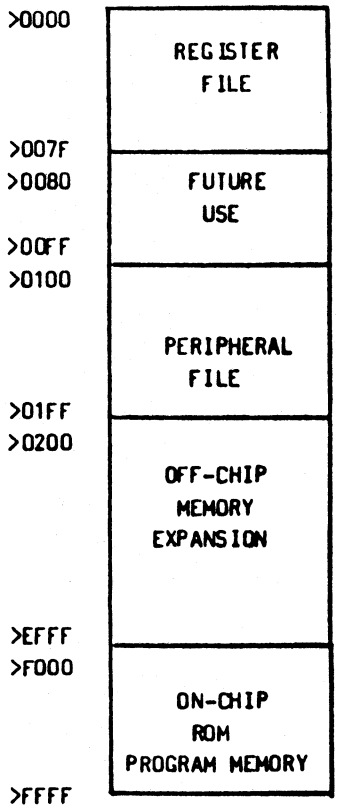
The PIC7000 Series has three machine registers accessible to the user; Program Counter, Status Register, and Stack Pointer. The 16 bit Program Counter (PC) contains the address of the next instruction to be executed. This address is referenced by the processor to fetch the next instruction from memory and is then automatically incremented. The Status Register (ST) has three conditional status bits, as well as a currentstate conditional bit, all testable under program control. The ST also includes the current state of the interrupt enable flag. The 8 bit Stack Pointer (SP) points to the last or top entry on the data stack located in the Register File (RF). This data/program stack provides a multilevel subroutine and interrupt capability, in addition to temporary data storage. The SP is automatically incremented just before data is pushed on the stack, and automatically decremented immediately after data is popped from the stack. There are 128 user-accessible registers in the RF. Register R0 and R1, the first two locations in the RF, are also referred to as registers A and B, respectively.

The PIC7001/7041 Memory Map is shown below. The 128 registers of on-chip RF are memory-mapped in order to permit use of the index and indirect addressing modes to access registers and allow use of 1 byte instructions. The RF is mapped from location >0000 to location >007F. The on-chip I/O ports are memory-mapped into a general memory space adjacent to the RF, known as the Peripheral File (PF). This permits use of the extended addressing modes to communicate with peripherals, in addition to the dedicated PF instructions.

PIC7000 SERIES MEMORY MAP



PIC7001



PIC7041

NOTE: General Instrument Microelectronics reserves the first 6 bytes of ROM on the PIC7041. Addresses from >F000 to >F005 on the PIC7040 may not be defined by the user program.

STATUS REGISTER

The Status Register contains three status bits (C,N, and Z) and the current interrupt enable bit (I) value, as shown:

7	6	5	4	3	2	1	0
C	N	Z	I	FUTURE USE			

C - CARRY OUT
N - SIGN BIT
Z - ZERO
I - INTERRUPT ENABLE

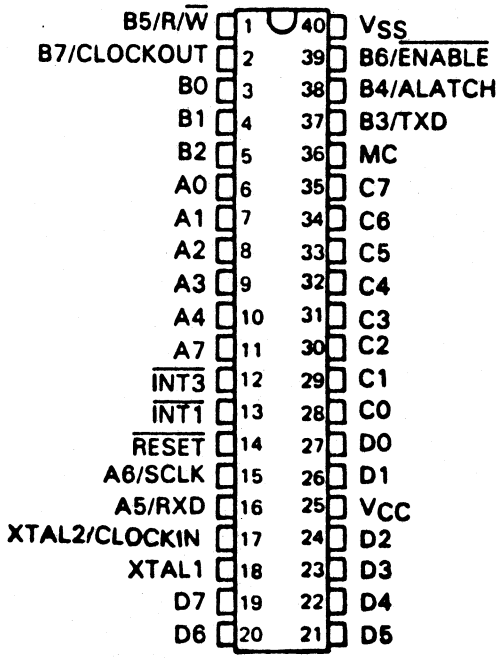
The three conditional status bits (C,N, and Z) are used for conditional branching. The interrupt enable bit determines whether or not mask programable interrupts are enabled. If the interrupt enable bit is equal to one, then the mask programable interrupts are enabled; otherwise, they are disabled. The interrupt enable bit is equal to zero immediately following the reset sequence. The interrupt enable bit can be individually set or cleared using the EINT or DINT instructions.

Single Chip, Peripheral Expansion, Full Expansion, and Microprocessor Modes

Signal	Function
A0 - A4, A7	I/O Port A: General purpose bidirectional I/O lines
A5 - A6/SCLK	I/O Port A: Input lines
B0 - B7	I/O Port B: Output lines
C0 - C7	I/O Port C: General purpose bidirectional lines
D0 - D7	I/O Port D: General purpose bidirectional lines
$\overline{INT1}$	Maskable interrupt
$\overline{INT3}$	Maskable interrupt
$\overline{RESET}$	$\overline{RESET}$
MC	Memory Control
XTAL2/CLKIN	Crystal input for control of internal OSC, input pin for external OSC or LRC networks
XTAL1	Crystal input for control of internal OSC, leave open for external OSC
V_{CC}	Supply voltage (+5V)
V_{SS}	Ground reference

PIN CONFIGURATION
40 LEAD DUAL IN LINE

(TOP VIEW)



BASIC INSTRUCTION SET SUMMARY

Mnemonic	Meaning	Status Bits Affected	Description
Single Operand Instructions			
CLR	Clear Operand	5,6,7	0 → dest
DEC	Decrement	5,6,7	Dest - 1 → dest
DECD	Decrement Double	5,6,7	Register pr - 1 → register pr
INC	Increment	5,6,7	Dest + 1 → dest
INV	Invert	5,6,7	Dest → dest
POP	Pop from Stack	5,6,7	Top of stack → dest, SP - 1 → SP*
PUSH	Push on Stack	5,6,7	SP + 1 → SP, Dest → top of stack
RL	Rotate Left	5,6,7	bn → bn+ 1, B7 → b0, C
RLC	Rotate Left through Carry	5,6,7	bn → bn+1, C → b0, b7 → C
RR	Rotate Right	5,6,7	bn+1 → bn, b0 → b7, C
RRC	Rotate Right through Carry	5,6,7	bn+1 → bn, C → b7, b0 → C
XCHB	Exchange with Register B	5,6,7	B → dest, N, Z set on Dest contents
Dual Operand Instructions			
ADC	Add with Carry	5,6,7	Source + dest + carry → dest
ADD	Add Bytes	5,6,7	Source + dest → dest
AND	AND bytes	5,6,7	Source logically ANDed with dest → dest
ANDP	AND Peripheral File	5,6,7	Source logically ANDed with PF → PF
CMP	Compare	5,6,7	Dest - source computed but not stored
DAC	Decimal Add w/Carry	5,6,7	Source + dest + carry → dest
DSB	Decimal Subtract w/Borrow	5,6,7	Dest - source - 1 + carry → dest
MOV	Move	5,6,7	Source → dest
MOVD	Move Double	5,6,7	Register pair → register pair
MOVP	Move to/from PF	5,6,7	Read or write data to PF
MPY	Multiply	5,6,7	Source register pr → register pr
OR	OR	5,6,7	Source logically ORed with dest → dest
ORP	OR Peripheral File	5,6,7	Source logically ORed with PF → PF
SBB	Subtract w/Borrow	5,6,7	Dest - source - 1 + carry → dest
SUB	Subtract Bytes	5,6,7	Dest - source → dest
XOR	Exclusive OR	5,6,7	Source exclusively ORed with dest → dest
XORP	Exclusive OR PF	5,6,7	Source exclusively ORed with PF → PF

*Destination may also be status register.

BASIC INSTRUCTION SET SUMMARY

Mnemonic	Meaning	Status Bits Affected	Description
Jump Instructions			
BTJO	Bit Test Jump If One	5,6,7	If source ANDed with dest 0, jump
BTJOP	Bit Test Jump If One PF	5,6,7	If source ANDed with PF 0, jump
BTJZ	Bit Test Jump If Zero	5,6,7	If source ANDed with Dest ≠ 0, jump
BTJZP	Bit Test Jump If Zero PF	5,6,7	If source ANDed with PF ≠ 0, jump
DJNZ	Dec. Reg. Jump Non-Zero	none	Dest - 1 → dest, if dest 0, jump
JMP	Jump Unconditional	none	PC + offset → PC
JC/JHS	Jump If Carry Set/ Jump If Higher or Same	none	If C = 1, PC + offset → PC
JN	Jump If Negative	none	If N = 1, PC + offset → PC
JNC/JL	Jump If No Carry/ Jump If Lower	none	If C = 0, PC + offset → PC
JNZ/JNE	Jump If Not Zero/ Jump If Not Equal	none	If Z = 0, PC + offset → PC
JP	Jump If Positive	none	If N = 0, Z = 0, PC + offset → PC
JPZ	Jump If Positive or Zero	none	If N = 0, PC + offset → PC
JZ/JEQ	Jump If Zero/ Jump If Equal to	none	If Z = 1, PC + offset → PC
Extended Instructions			
BR	Branch	none	Dest → PC
CMPA	Compare to A Register	5,6,7	A - Source computed
LDA	Load A Register	5,6,7	Source → A
STA	Store A Register	5,6,7	A → dest
Control Instructions			
CLRC	Clear Carry Bit	5,6,7	0 → C, N, Z, set from A register
DINT	Disable Interrupts	4,5,6,7	0 → I, 0 → C, 0 → N, 0 → Z
EINT	Enable Interrupts	4,5,6,7	1 → I, 1 → C, 1 → N, 1 → Z
IDLE	Idle until Interrupt	none	Suspend until Interrupt
LDSP	Load Stack Pointer	none	B register → SP
NOP	No Operation	none	PC + 1 → PC
SETC	Set Carry	5,6,7	1 → C, 0 → N, 1 → Z
STSP	Store Stack Pointer	none	SP → B register
SWAP	Swap Nibbles	5,6,7	B7-B4 ↔ B3-B0
TSTA	Test A Register	5,6,7	0 → C, N, Z, set from A register
TSTB	Test B Register	5,6,7	0 → C, N, Z, set from B register

NOTE:
Some conditional Jump instructions have two names: one indicating the condition of the Status Register Bits that are tested and one indicating the result of a CMP (Compare) instruction.

BASIC INSTRUCTION SET SUMMARY

Mnemonic	Meaning	Status Bits Affected	Description
Subroutine Instructions			
CALL	Call Subroutine	none	SP + 1→SP, PCMS byte →stack SP + 1→SP, PCLS byte →stack Operand address→PC
RETI	Return from Interrupt	loaded from Stack	Stack→PCLS byte, SP - 1→SP Stack→PCMS byte, SP - 1→SP Stack→ST, SP - 1→SP
RETS	Return from Subroutine	none	Stack→PCLS byte, SP - 1→SP Stack→PCMS byte, SP - 1→SP
TRAP 0 • • TRAP 23	Trap to Subroutine	none	SP + 1→SP, PCMS byte→stack SP + 1→SP, PCLS byte→stack entry vector→PC

NOTE: Some conditional jump instructions have two names: one indicating the condition of the Status Register bits that are tested, and one indicating the result of a CMP (compare) instruction.

STANDARD INSTRUCTION SET

TERMS AND DEFINITIONS

SYMBOL	MEANING
$	Current value of the program counter
A	Register A (also equal to R0)
B	Register B (also equal to R1)
Rx	Register x of Register File (0< x <127)
Px	Port x of Peripheral File (0< x <255)
PC	Program Counter
ST	Status Register

ADDRESSING MODES

Register A Addressing

Register A addressing specifies that the A register (R0) contains the operand. It should be noted that the A register address is implicity specified by the opcode and thus does not require an address field byte similar to the register file addressing mode.

DEC A Decrement the contents of A.

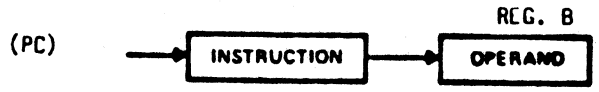
REGISTER A ADDRESSING

Register B Addressing

Register B addressing specifies that the B register (R1) contains the operand. It should be noted that the B register address is implicity specified by the opcode and thus does not require an address field byte similar to the register file addressing mode.

```
INC     B       Increment the contents of B.
MOV     A,B     Copy the contents of A into B.
```

REGISTER B ADDRESSING

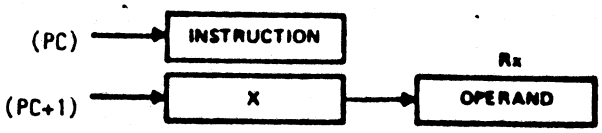


Register File Addressing

Register File (RF) addressing specifies a register that contains the operand. The register number is specified by a one-byte field in the multi-byte instruction. The register number can be written as Rx where x is the register number or as simply x.

```
ADD     R113        Replace contents of A with sum of contents of A and R113.
MOV     R5, R78     Copy contents of R5 to R78.
```

REGISTER FILE ADDRESSING Rx



It should be noted that the register file addressing may be used to address the A and B registers since these registers are also in the register file (R0 and R1). Consequently, the following example adds the contents of R42 to the contents of the A register:

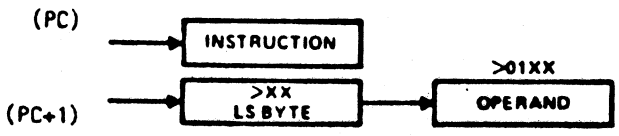
```
LABEL     ADD     R42, R0
```

Peripheral File Addressing

Peripheral File (PF) addressing specifies a PF location that contains the operand. The PF port number is specified by a one-byte field in the multi-byte instruction. The PF typically contains the I/O ports and status/control registers. The port number is written as Px where x is the port number, or as simply x when using peripheral instructions.

```
ANDP    A,P3        Replace contents of P3 with logical AND of contents of A and P3.
MOVP    P19,B       copy contents of P19 to B.
```

PERIPHERAL FILE ADDRESSING Px



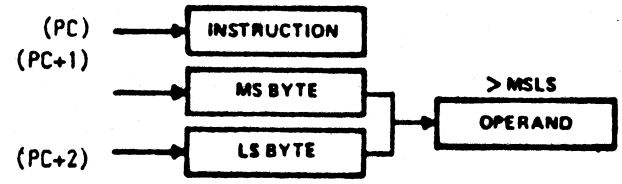
NOTE: All write instructions using the Peripheral File addressing mode perform a read before a write operation to the Peripheral File location.

Direct Memory Addressing

Direct Memory addressing specifies a 16-bit address that contains the operand. A direct memory address is written as an expression preceded by an "at" sign, (@).

DIRECT MEMORY ADDRESSING @ LABEL

```
LDA     @>F47D      Copy contents of memory location
*                   >F47D to register A.
BR      @THERE      Branch to location THERE.
```

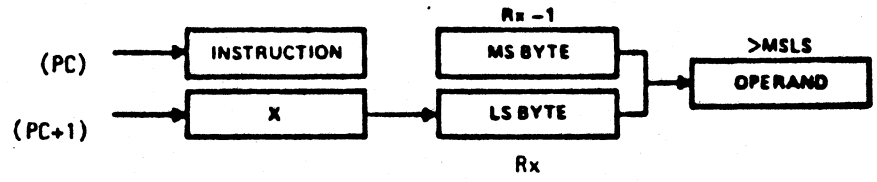


Register File Indirect Addressing

Register file (RF) indirect addressing specifies the address of a pair of register file locations that contain the address of the operand. An indirect RF address is written as a register name preceded by an asterisk (*). The specified register contains the least-significant half of the 16-bit address. The most-significant half of the address is contained in the next lower register. If R0 is the specified register, then the most-significant half of the address would be contained in R255, which is not implemented. Therefore, when using RF indirect addressing, R0 cannot be the specified register.

REGISTER FILE INDIRECT ADDRESSING *Rx

```
        STA     *R46        Copy the contents of register A
*                           into address >43F8.
*                           (contents of R45 and R46, are >43
*                           and >F8, respectively)
```

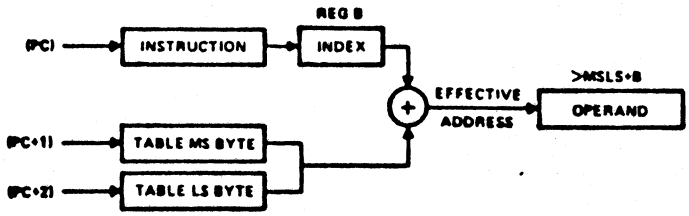


Indexed Addressing

Indexed Addressing specifies a memory address that contains the operand. The address is the sum of the contents of the B register and a 16-bit direct address. An indexed address is written as an expression preceded by an @ and followed by (B). Any carry is rippled into the MSB.

INDEXED ADDRESS @ TABLE (B)

```
*       STA     @TABLE(B)       Copy the contents of register A into location
*                               specified by the contents of register B and
*                               value of symbol TABLE.
```

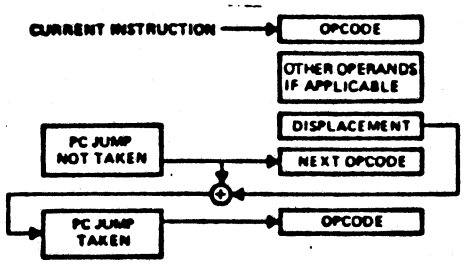


Program Counter Relative Addressing

Program Counter (PC) relative addressing is used by the jump instructions. The assembler evaluates the expression and then the program instruction location immediately following the current instruction. This calcuated value must be in the range of -128 through +127. When the instruction is in relocatable code (i.e., when the location counter is relocatable), then the relocation type of the evaluated expression must match the relocation type of the current location counter. When the instruction is in absolute code, the expression must be absolute.

PROGRAM COUNTER RELATIVE JUMP

```
*       JNC     HERE            Jump to HERE if the carry status
                                bit is equal to zero.
*       BTJO    R12,R117,HERE   Jump to HERE if a bit in R12 and the
                                corresponding bit in R117 are equal to one.
```

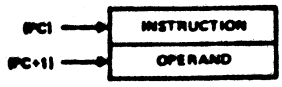


Immediate Addressing

Immediate Addressing instructions use the contents of a byte following the opcode byte as an operand. The immediate value is an expression, and the value of the expression is placed in a byte following the opcode byte. The immediate operand is written as an expression preceded by a percent sign (%).

IMMEDIATE ADDRESSING

```
        MOV     %>98,R123       Replace the contents of R123
*                               with >98.
        ANDP    %MASK,P10       Logically AND the value of MASK
*                               and the contents of P10; copy the
*                               results to P10.
```



INSTRUCTION EXECUTION TIMES

OPERATION		ADDRESSING MODES								NOTES
		A	B	RF	PF	@lab	*RF	@lab(B)	OTHER	
ADC	B,--	1/5								
	RF,--	2/8	2/8	3/10						
	%iop,--	2/7	2/7	3/9						
ADD	B,--	1/5								
	RF,--	2/8	2/8	3/10						
	%iop,--	2/7	2/7	3/9						
AND	B,--	1/5								
	RF,--	2/8	2/8	3/10						
	%iop,--	2/7	2/7	3/9						
ANDP	A,--				2/10					
	B,--				2/9					
	%iop,--				3/11					
BTJO	B,--	2/7								
	RF,--	3/10	3/10	4/12						
	%iop,--	3/9	3/9	4/11						
BTJOP	A,--				3/11					(1)
	B,--				3/10					
	%iop,--				4/12					
BTJZ	B,--	2/7								
	RF,--	3/10	3/10	4/12						
	%iop,--	3/9	3/9	4/11						
BTJZP	A,--				3/11					(1)
	B,--				3/10					
	%iop,--				4/12					
BR	--					3/10	2/9	3/12		
CALL	--					3/14	2/13	3/16		
CLR	--	1/5	1/5	2/7						
CLRC	--								1/6	
CMP	B,--	1/5								
	RF,--	2/8	2/8	3/10						
	%iop,--	2/7	2/7	3/9						
CMPA	--					3/12	2/11	3/14		
DAC	B,--	1/7								
	RF,--	2/10	2/10	3/12						
	%iop,--	2/9	2/9	3/11						
DEC	--	1/5	1/5	2/7						
DECD	--	1/9	1/9	2/11						
DINT	--								1/5	
DJNZ	--	2/7	2/7	3/9						(1)
DSB	B,--	1/7								
	RF,--	2/10	2/10	3/12						
	%iop,--	2/9	2/9	3/11						
EINT	--								1/5	
IDLE	--								1/6+	
INC	--	1/5	1/5	2/7						
INV	--	1/5	1/5	2/7						
JMP	--								2/7	
Jcnd	tab--								2/5	(1,2)

INSTRUCTION EXECUTION TIMES (continued)

OPERATION		ADDRESSING MODES								NOTES
		A	B	RF	PF	@lab	*RF	@lab(B)	OTHER	
LDA						3/11	2/10	3/13		
LDSP									1/5	
MOV	A,--		1/6	2/8						
	B,--	1/5		2/7						
	RF,--	2/8	2/8	3/10						
	%iop,--	2/7	2/7	3/9						
MOVD	%iop,--			4/15						
	%iop(B),--			4/17						
	RF,--			3/14						
MOVP	A,--				2/10					
	B,--				2/9					
	%iop,--				3/11					
	PF,--	2/9	2/8							
MPY	B,--	1/44								
	RF,-- --	2/47	2/47	3/49						
	%iop,--	2/46	2/46	3/48						
NOP	--								1/4	
OR	B,--	1/5								
	RF,--	2/8	2/8	3/10						
	%iop,--	2/7	2/7	3/9						
ORP	A,--				2/10					
	B,--				2/9					
	%iop,--				3/11					
POP	--	1/6	1/6	2/8						
POP ST	--								1/6	
PUSH	--	1/6	1/6	2/8						
PUSH ST	--								1/6	
RETI	--								1/9	
RETS	--								1/7	
RL	--	1/5	1/5	2/7						
RLC	--	1/5	1/5	2/7						
RR	--	1/5	1/5	2/7						
RRC	--	1/5	1/5	2/7						
SBB	B,--	1/5								
	RF,--	2/8	2/8	3/10						
	%iop,--	2/7	2/7	3/9						
SETC	--								1/5	
STA	--					3/11	2/10	3/13		
STSP	--								1/6	
SUB	B,--	1/5								
	RF,--	2/8	2/8	3/10						
	%iop,--	2/7	2/7	3/9						
SWAP	--	1/8	1/8	2/10						
TSTA	--								1/6	
TSTB	--								1/5	
TRAPn	--								1/14	
XCHB	--	1/6	1/6	2/8						
XOR	B--	1/5								
	RF,--	2/8	2/8	3/10						
	%iop,--	2/7	2/7	3/9						

INSTRUCTION EXECUTION TIMES (continued)

OPERATION	ADDRESSING MODES A	B	RF	PF	@lab	*RF	@lab(B)	OTHER	NOTES
XOPR A,--				2/10					
B,--				2/9					
%iop,--				3/11					

NOTES:
1. Add 2 to cycle count if branch is taken.
2. Conditional Jump Instructions.

NOTATION: Data Form - number of bytes/number of internal clock cycle.
A A register
B B register
RF Register file number
PF Peripheral file number
lab Label
iop Immediate operand

OPCODES (HEXADECIMAL) BY INSTRUCTION

	SINGLE OPERAND			DUAL OPERAND									PERIPHERAL					EXTENDED			OTHER	STATUS WORD	
	A	B	Rn	A,B	B,A	Rn,A	%n,A	Rn,B		%nRm	A,Rn	B,Rn	A,Pn	Pn,A	B,Pn	Pn,B	%n,Pn	DIRECT	INDIRECT	INDEXED	OTHER	COND. BITS	INT. EN. BIT
ADC					69	19	29	35	59	49	79											X	
ADD					68	18	28	38	58	48	78											X	
AND					63	13	23	33	53	43	73											X	
ANDP													83		93		A3					X	
BTJO					66	16	26	36	56	46	76											X	
BTJOP													86		96		A6					X	
BTJZ					67	17	27	37	57	47	77											X	
BTJZP													87		97		A7					X	
BR																		8C	9C	AC			
CALL																		8E	9E	AE			
CLR	B5	C5	D5																			X	
CLRC																					80	X	
CMP					6D	1D	2D	3D	5D	4D	7D											X	
CMPA																		8D	9D	AD		X	
DAC					6E	1E	2E	3E	5E	4E	7E											X	
DEC	82	C2	D2																			X	
DECD	B8	C8	D8																			X	
DINT																					06	X	X
DJNZ	BA	CA	DA																				
DSB					6F	1F	2F	3F	5F	4F	7F											X	
EINT																					05	X	X

OPCODES (HEXADECIMAL) BY INSTRUCTION (continued)

	SINGLE OPERAND			DUAL OPERAND										PERIPHERAL					EXTENDED			OTHER	STATUS WORD	
	1 A	2 B	3 Rn	4 A,B	5 B,A	6 Rn,A	7 #n,A	8 Rn,B	9	10	11 #nRn	12 A,Rn	13 B,Rn	14 A,Pn	15 Pn,A	16 B,Pn	17 Pn,B	18 #n,Pn	19 DIRECT	20 INDIRECT	21 INDEXED	OTHER	COND. BITS	INT. EN. BIT
IDLE																						01		
INC	B3	C3	D3																				X	
INV	B4	C4	D4																				X	
JMP																						E0		
JC/JHS																						E3		
JN																						E1		
JNC/JL																						E7		
JNZ/JNE																						E6		
JP																						E4		
JPZ																						E5		
JZ/JEQ																						E2		
LDA																			8A	9A	AA		X	
LDSP																						0D		
MOV				C0	62	12	22	32	52	42	72	D0	D1										X	
MOVD																			88	98	A8		X	
MOVP														82	80	92	91	A2					X	
MPY					6C	1C	2C	3C	4C	5C	7C												X	
NOP																						00		
OR					64	14	24	34	54	44	74												X	
ORP														84		94		A4					X	
POP	89	C9	D9																			08	X	
PUSH	88	C8	D8																			0E	X	
RETI																						08		
RETS																						0A		
RL	BE	CE	DE																				X	
RLC	BF	CF	DF																				X	
RR	BC	CC	DC																				X	
RRC	BD	DC	DD																				X	
SBB					6B	1B	2B	3B	5B	4B	7B												X	
SETC																						07	X	
STA																			8B	9B	AB		X	
STSP																						09		
SUB					6A	1A	2A	3A	5A	4A	7A												X	
SWAP	B7	C7	D7																				X	
TSTA																						80	X	
TSTB																						C1	X	
TRAP																						EBFF		
XCHB	B6		D6																				X	
XOR					65	15	25	35	55	45	75												X	
XORP														85		95		A5					X	

INTERRUPTS AND RESET

The PIC7001/7041 implements six hardware interrupt levels. Level 0 is the highest priority and is reserved for the RESET function. Level 1 follows and is a user-defined external interrupt (INT1), level 2 is reserved for the first hardware timer, level 3 is another user-defined external interrupt (INT3), Level 4 is the serial port interrupt (INIT4) and level 5 is reserved for the second hardware timer.

The external interrupt interface consists of three discrete input lines (RESET, INT1, INT3). Interrupts are asserted by holding a low level on these pins and require no external synchronization. Interrupt levels 2 and 5 are asserted upon rollover of the programmable timer. All external interrupts and RESET have Schmitt Trigger inputs.

RESET must be held active (low) for a minimum of 1.25 internal machine clock frequencies (ϕ) to guarantee recognition. During assertion of RESET, Ports A, C and D are placed in the high impedance input mode, and all bits of PORT are put to ones regardless of the state of the oscillator. When RESET is removed, the following operations are performed prior to the first instruction acquisition.

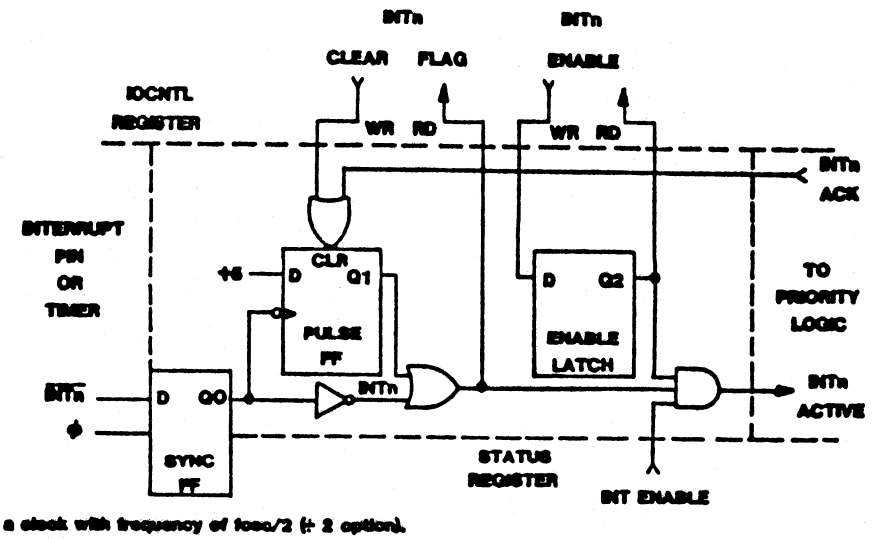
a) All zeros (>00) are written to the internal I/O Control Register 0 (IOCNTL0) and Status Register (ST). This disables all interrupts globally, and INT1, INT2, AND INT3 individually. Note that since I/O Control Register 1 (IOCNTL1) is not written during RESET it should be written to immediately after RESET to ensure that INT4 and INT5 are individually disabled.
b) The PC value just before RESET is stored in the register pair R0 and R1, and the SP is initialized to 01.
c) The RESET vector is fetched from >FFFE and >FFFF and placed in the PC.
d) The serial port registers are reset.

Program execution begins at the address placed in the PC from the level 0 interrupt vector location. Register pair, R0 and R1, contain the PC at the time of the RESET interrupt; SP has been initialized to >01, and programmable I/O pins (A0-A4, A7, C7-C0, D7-D0) are in the input mode. Since the SP addresses the last value pushed, the data stack may now grow from address >02 onward. RESET (interrupt level 0) cannot be masked and will be recognized immediately, even in the middle of an instruction.

The PIC7001/7041 implements a very flexible prioritized interrupt structure with individually masked interrupt levels. Multiple devices may be connected on the same interrupt line by wire-ORing their interrupt request lines onto the INT1 or INT3 input lines. Interrupt inputs may be a level or a pulse, and may be tested without actually recognizing the interrupt, permitting flexible multidevice control. Under program control, each interrupt routine may retain complete control of the processor, or allow nested interrupts.

The figure below shows the interrupt logic for each of the five maskable interrupt levels. INT1 and INT3 are applied externally onto input pins, while INT2 is applied internally each time the first timer decrements past zero. INT5 is internally activated every time the second timer decrements past zero, and INT4 is applied when the serial port is ready for data transfer.

INTERRUPT LOGIC



NOTE: ϕ is a clock with frequency of fosc/2 (÷ 2 option), fosc/4 (÷ 4 option).

There are 19 machine cycles (t_c) required between the end of an instruction in the interrupted program and the start of the first instruction of the interrupt routine. Interrupting out of idle requires 17 machine cycles.

RESET/INTERRUPT VECTOR LOCATION

VECTOR MSB	VECTOR LSB	DESCRIPTION	SERVICE ORDER
>FFFE	>FFFF	RESET	Immediate
>FFFC	>FFFD	INT1(external)	1
>FFFA	>FFFB	INT2(timer)	2
>FFF8	>FFF9	INT3(external)	3
>FFF6	>FFF7	INT4(serial port,timer 3)	4
>FFF4	>FFF5	INT5(timer 2)	5

Memory Expansion Modes

All PIC7001/7041 devices feature 32 pins which can be used for general purpose I/O. Several of the pins may be reconfigured to form an off-chip memory expansion bus, allowing the microcomputer to reference up to 64K bytes of ROM and RAM. Four memory expansion modes can be used to configure the part to meet a variety of application requirements. These modes are described in the following paragraphs.

Single-Chip Mode:

All 32 lines are used for input/output in the single-chip mode, and no off-chip memory bus in implemented. All program and data reside in the on-chip ROM and RAM.

Peripheral Expansion Mode:

In this mode, 20 lines are used for input/output, and 12 implement an 8-bit address bus to external memory. In this mode, an additional 238 external addresses are available for ROM, RAM, or peripheral devices.

Full Expansion Mode:

In the Full Expansion mode, 12 lines are available for input/output, and 20 implement a 16-bit address bus, as well as a multiplexed 8-bit data bus, to external memory. The on-chip ROM is still used, but additional off-chip memory for additional program or data storage may be referenced as well.

Microprocessor Mode:

As in the Full Expansion mode, 12 lines are available for input/output and 20 for a 16-bit address bus in addition to a multiplexed 8-bit data bus. Addresses for the on-chip ROM, however, are available off-chip, allowing the program to be prototyped in EPROM.

The Table below summarizes the memory expansion configuration options.

MEMORY EXPANSION CONFIGURATION OPTIONS

MODE	ON-CHIP RAM BYTES* 7001/7041	ON-CHIP ROM BYTES* 7001/7041	ON-CHIP I/O LINES* 7001/7041	OFF-CHIP MEMORY BYTES* 7001/7041
Single-Chip	na/128	na/4096	na/32**	na/0/0
Peripheral Expansion	na/128	na/4096	na/20**	na/246
Full Expansion	na/128	na/4096	na/12**	na/61168
Microprocessor	128	0	12/12**	65262/65262

* 7001 cannot be configured in the single-chip, peripheral expansion, or full expansion mode.
** When using the Serial Port, two I/O lines are used for RxD and TxD.

The 32 I/O pins are organized as four 8-bit ports, A, B, C, and D. Operation of these I/O ports, controlled by registers in the Peripheral File, is as follows:

Port A I/O lines A0-A4 and A7 are bidirectional. Pins A6 and A7 may also be used to clock the on-chip Timer/Event Counters, Timer 1 and Timer 2. I/O lines A5 and A6 are input only and are multifunctional. Pin A5 is RXD which receives incoming serial data and pin A6 is the serial clock input or output when using the serial port.

Port B Port B is always used as an output port, providing at least four general purpose output lines on the least significant nibble (pins B3-B0). When an off-chip memory bus is implemented, the most significant four bits (pins B7-B4) provide bus control signals. When the Single-Chip mode is in effect, pins B7-B4 provide four additional general purpose outputs. Pin B3 is also the serial output line (TXD) for the serial port.

Port C In Single-Chip mode, Port C is a bidirectional port where any of its eight bits may be programmed as an input or output bit under software control. In any other mode, Port C becomes a multiplexed address/data port for the off-chip memory bus. The least significant byte of a 16-bit address is provided in this case.

Port D In Single-Chip or Peripheral Expansion mode, Port D is a bidirectional port where any of its eight bits may be programmed as an input or output bit under software control. In Full Expansion and Microprocessor mode, Port D provides the most significant byte of the 16-bit address.

The memory expansion mode for the PIC7000 family devices is selected with a combination of hardware and software. An external Mode Control (MC) input pin, if at a logic one, will force the 7001 or 7041 into the Microprocessor mode. If MC is held at a logic zero, the remaining memory expansion modes are selected by software, and are only used by the and 7041. Operation of the 7001 when the MC is held at logic zero is undefined.

The two MSBs of the I/O Control Register, IOCNTL0 (7-6), encode the memory expansion mode. At reset, the IOCNTL0 register is initialized by hardware to all zeroes. Initialization software may then modify the IOCNTL0 register to set up the desired memory expansion mode. The mode select conditions for the various memory expansion modes are outlined in the tables below.

CONDITIONS FOR MEMORY EXPANSION MODES

	MODE SELECT CONDITIONS		
MODE	MODE CNTL PIN	I/O CONTROL REG. BIT 7,6	
Single-Chip	0V	0	0
Peripheral Expansion	0V	0	1
Full Expansion	0V	1	0
Microprocessor	V_{CC}	X	X

NOTE: X = Don't Care

PERIPHERAL FILE MEMORY MAP - SINGLE CHIP MODE

I/O PORT	MEMORY ADDRESS*	I/O REGISTER
P0	>0100	I/O Control 0 (IOCNTL0)
P1	>0101	Reserved
P2	>0102	Timer Data (T1DATA)
P3	>0103	Timer Control (T1CTRL)
P4	>0104	Port A Data Value (APORT)
P5	>0105	Port A Data Direction (ADDR)
P6	>0106	Port B Data Value (BPORT) (Output Only)
P7	>0107	Reserved
P8	>0108	Port C Data Value (CPORT) (I/O)
P9	>0109	Port C Data Direction (CDDR)
P10	>010A	Port D Data Value (DPORT) (I/O)
P11	>010B	Port D Data Direction (DDDR)
P12-P15	>010C - >010F	Not Available
P16	>0110	I/O Control 1 (IOCNTL1)
P17	>0111	Serial Mode (SMODE)/Serial Control 0 (SCTL0)/Status Register (SSTAT)
P18	>0112	Timer 2 Data (T2DATA)
P19	>0113	Timer 2 Control (T2CTRL)
P20	>0114	Timer 3 Data (T3DATA)
P21	>0115	Serial Control "1" (SCTL1)
P22	>0116	Receiver Buffer (RXBUF)
P23	>0117	Transmitter Buffer (TXBUF)
P24-P255	>0118 - >01FF	Not Available

*Hexadecimal

PERIPHERAL FILE MEMORY MAP - PERIPHERAL EXPANSION MODE

I/O PORT	MEMORY ADDRESS	I/O REGISTER
P0	>0100	I/O Control (IOCNTL0)
P1	>0101	Reserved
P2	>0102	Timer 1 Data (T1DATA)
P3	>0103	Timer 1 Control (T1CTRL)
P4	>0104	Port A Data Value (APORT)
P5	>0105	Port A Data Direction (ADDR)
P6	>0106 B3-B0	Port B Data Value (BPORT) (See Below)
P6	>0106 B7-B4	Peripheral File Expansion
P7	>0107	Reserved
P8-P9	>0108 - >0109	Peripheral File Expansion
P10	>010A	Port D Data Value (DPORT) (I/O)
P11	>010B	Port D Data Direction (DDDR)
P12-P15	>010C - >010F	Peripheral File Expansion
P16	>0110	I/O Control 1 (IOCNTL1)
P17	>0111	Serial Mode (SMODE)/Serial Control 0 (SCTL0)/Status Register (SSTAT)
P18	>0112	Timer 2 Data (T2DATA)
P19	>0113	Timer 2 Control (T2CTRL)
P20	>0114	Timer 3 Data (T3DATA)
P21	>0115	Serial Control 1 (SCTL1)
P22	>0116	Receiver Buffer (RXBUF)
P23	>0117	Transmitter Buffer (TXBUF)
P24-P255	>0118 - >01FF	Peripheral File Expansion

PERIPHERAL FILE MEMORY MAP - FULL EXPANSION AND MICROPROCESSOR MODES

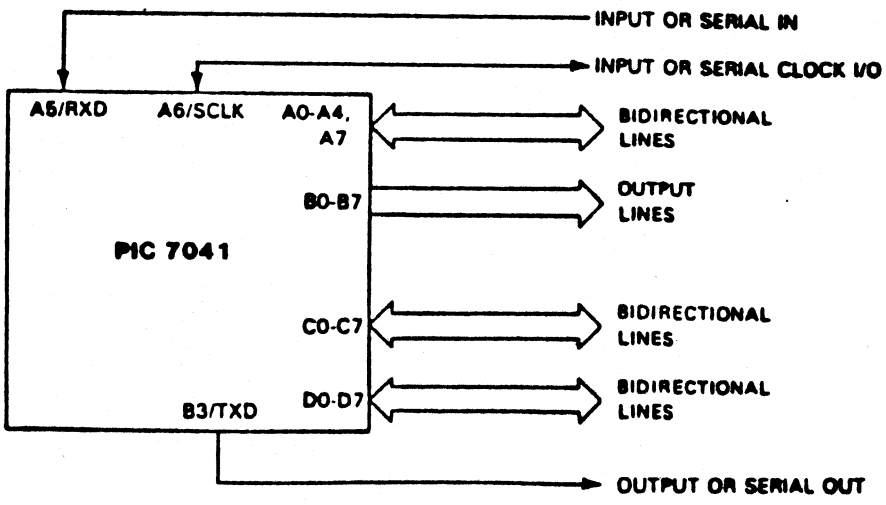
I/O PORT	MEMORY ADDRESS	I/O RESISTER
P0	>0100	I/O Control (IOCNTL0)
P1	>0101	Reserved
P2	>0102	Timer 1 Data (T1DATA)
P3	>0103	Timer 1 Control (T1CTRL)
P4	>0104	A Port Data Value (APORT)
P5	>0105	Port A Data Direction (ADDR)
P6	>0106 B3-B0	B Port Data Value (BPORT) (Same as Peripheral Expansion mode)
P6	>0106 B7-B4	Peripheral File Expansion
P7	>0107	Reserved
P8-P15	>0108 - >010F	Peripheral File Expansion
P16	>0110	I/O Contjrol 1 (IOCNTL1)
P17	>0111	Serial Mode (SMODE)/Serial Control 0 (SCTL0)/Status Register (SSTAT)
P18	>0112	Timer 2 Data (T2DATA)
P19	>0113	Timer 2 Control (T2CTRL)
P20	>0114	Timer 3 Data (T3DATA)
P21	>0115	Serial Control (SCTL1)
P22	>0116	Receiver Buffer (RXBUF)
P23	>0117	Transmitter Buffer (TXBUF)
P24-P255	>0118 - >01FF	Peripheral File Expansion

SINGLE CHIP MODE

The on-chip I/O is configured at $\overline{\text{RESET}}$ to be in the Single-Chip mode with four external I/O ports. The two MSBs in the I/O control register (IOCNTL0) must be 0s to elect the Single-Chip mode. These bits are automatically cleared upon $\overline{\text{RESET}}$.

There are four 8-bit on-chip I/O ports. All four ports are located in the Peripheral File and may thus be used efficiently with the I/O operating instructions.

I/O PORTS - SINGLE-CHIP MODE



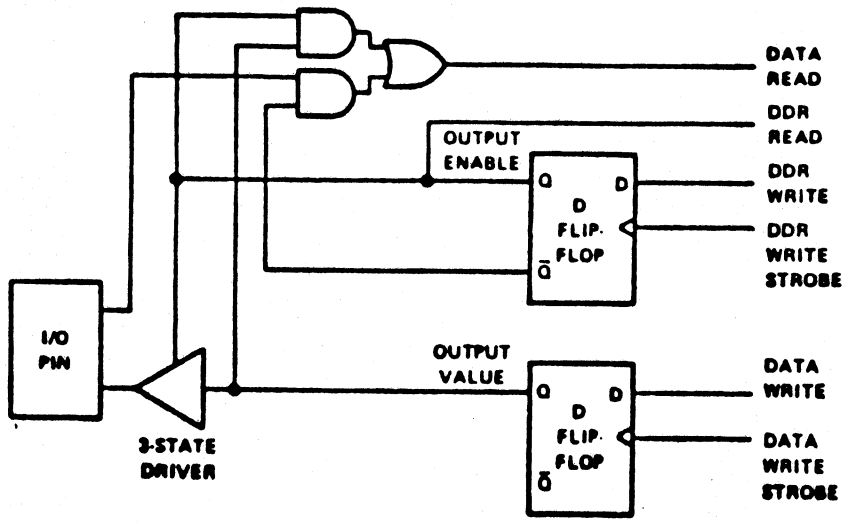
Port A is a 8-bit bidirectional input/output with the A0 line as the LSB and the A7 line as the MSB. Lines A0 to A4, and A7 are bidirectional bit programmable lines. The direction of these lines is controlled through the A Data Direction register (ADDR). Associated with each bidirectional port (A, C, D) is a data register and a DDR. Each I/O line can be programmed to be either an input or output. Setting a 1 in a DDR bit causes the corresponding I/O line to be an output. A zero causes the line to be a high impedance input. The data register is used to be read and write the I/O lines. A one in the data register will cause a high on the I/O line if it is programmed as an output by the DDR.

Writing to a bit position configured as an input bit has no effect on the value subsequently read from that bit. Line A5 is the A5/RXD and is used as the RXD by the serial port. Line A6 is an input only Port A line and can be used to input an external source for Timer 2. A6 is also used by the bidirectional serial clock (SCLK). An internal SCLK is output and an external SCLK is input on line A6 respectively.

When a serial port is not used, lines A5 and A6 are available as general purpose input lines. Line A7 is a bidirectional Port A line and can also be used as the external input for Timer 1 when it is in the external clock mode. Input only Port A lines are read through the A Data register. Reading the corresponding A DDR bits gives a 0. Writing to their corresponding A Data register and A DDR bits has no effect.

Port B is an 8-bit output only port. When Port B is read, the value of the pin is read, except for the B3 line. Line B3 is B3/TXD and is used as the TXD line by the serial port. When B3 is read, the value last written to B3 (stored in a data latch) is returned. The TXD signal has no effect on the B3 value read. Ports C and D are 8-bit bidirectional data ports in which each bit can be individually programmed to be an input or an output. The bidirectional I/O lines are illustrated below.

BIDIRECTIONAL I/O LINES



The following segment of code illustrates how to program lines D0-D2 to be output lines and lines D3-D7 to be input lines. The programmer can then pass output data through D0-D2 and read in input data on D3-D7 as well as the most recently output data on D0-D2.

```
DPORT   EQU     P10
DDDR    EQU     P11

        MOVP    >07,DDDR        Set I/O LINES
        ORP     >02,DPORT       Set OUTPUT BIT D1
        BTJOP   >80,DPORT,LABEL JUMP IF INPUT BIT D7 SET
```

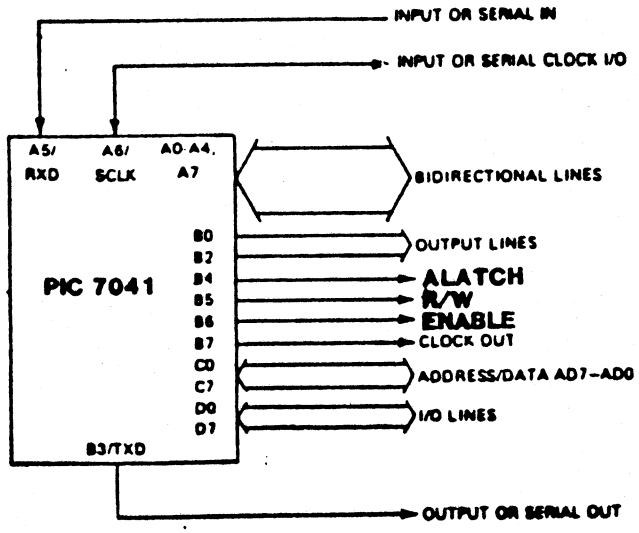
PERIPHERAL EXPANSION MODE

This mode greatly enhances the I/O capability of the 7041. In this mode, the 7041 configures itself to increase its I/O handling by giving the user external access to the Peripheral File (PF) address space. The PF is optimized for I/O operation. The 7000 Family has dedicated instructions to enhance I/O manipulation using the PF. There are 238 bytes of PF address available for expansion in this mode.

In the Peripheral Expansion mode, I/O lines C0-C7 and B4-B7 are used for an external memory interface. I/O lines C0 through C7 are used to pass multiple address/data and output lines B4-B7 are used for memory interface timing and control.

Port C Data Register (P8) and Port C DDR (P9) addresses become off-chip addresses along with the remainder of the Peripheral File address space (P24 through P255). Addresses P1, P5, P7 and P12-P15 remain reserved as in the Single-Chip mode. Port B is a special case as it is both on-and off-chip. The Port B low nibble (B0-B3) continues to function as output lines as in the Single-chip mode. In the Peripheral Expansion mode however, when a write is performed at Port B (6), the address and data are multiplexed off-chip through Port C. In addition, the low nibble of the output data is placed on the lower four output lines of Port B. The values of the B0-B3 data latches are read, as in the Single-Chip mode. The upper nibble is obtained through Port C from off-chip.

I/O PORTS - PERIPHERAL EXPANSION MODE

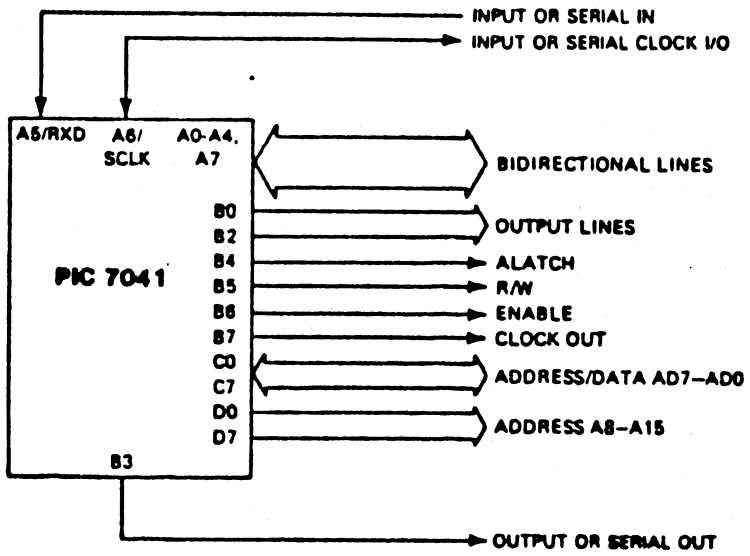


FULL EXPANSION AND MICROPROCESSOR MODES

The Full Expansion modes uses I/O lines C0-C7, D0-D7, and B4-B7 for an external memory interface. I/O lines D0 through D7 are used to pass the most significant byte of the address. I/O lines C0 through C7 are used to pass multiplexed address/ data.

Output lines B4 through B7 are used for memory interface timing and control. In Full Expansion mode, the Peripheral File registers are treated as in Peripheral mode, except that the Port D Data Register (P10) and Port D DDR (P11) become off-chip addresses.

I/O PORTS - FULL EXPANSION/MICROPROCESSOR MODES



The Microprocessor mode allows the PIC7000 family to operate as an 8-bit microprocessor. This mode functions the same way as the Full Expansion mode with the exception that there is no on-chip ROM program memory.

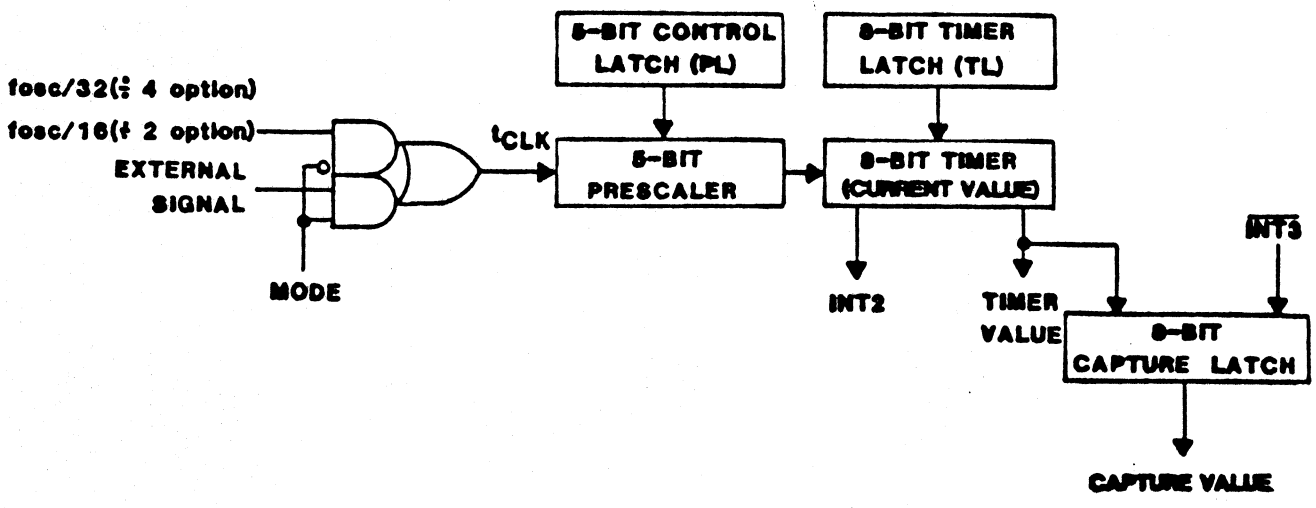
The I/O for the Microprocessor mode is the same as it is for the Full Expansion mode. The only difference between the two modes is the absence of on-chip ROM in the Microprocessor. Addresses in the range >F000 to >FFFF are treated as external memory locations.

PROGRAMMABLE TIMER/EVENT COUNTERS

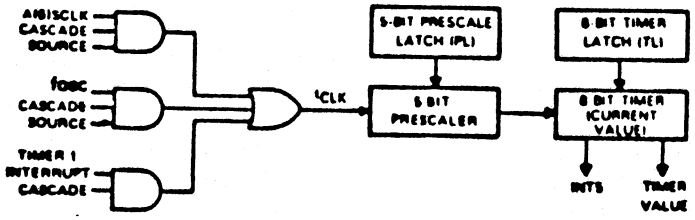
The Programmable Timer/Event Counters are 8-bit modulo-N counters with programmable prescaled clock sources, as shown in the diagrams below.

Timer data (P2, P18) and control (P3, P19) registers in the Peripheral File are described in the Table on the following page.

TIMER 1



TIMER 2



The clock source and prescaling value of both timers are determined by the timer control registers (T1CTRL/T2CTRL). These control bits are write-only and therefore restrict timer control register manipulations to the following instructions:

MOVP % immediate value, Pn	SIA @ Peripheral File Address
MOVP A, Pn	SIA * Rn (Rn-1 and Rn = Peripheral File Address)
MOVP B, Pn	SIA @ Peripheral File Address (B)

The same instructions are required for writing to the timing data registers.

The clock source of Timer 1 is determined by bit 6 (SOURCE) of T1CTRL. A SOURCE bit 0 selects the internally generated $f_{OSC}/32$ (-4 option, $f_{OSC}/8$) clock and places the Timer/Event Counter in the Real Time Clock (RTC) mode. A SOURCE bit of 1 selects the external clock source and places the Timer/Counter in the Event Counter mode. In the external mode, the clock sources for Timers 1 and 2 are input on the MSB of I/O port A (A7) and (A6) respectively.

TIMER DATA AND CONTROL REGISTERS

TIMER 1 DATA REGISTER - T1DATA

T DATA >0102

7	6	5	4	3	2	1	0	
CURRENT TIMER VALUE								READ
TIMER LATCH VALUE								WRITE

TIMER 1 CONTROL REGISTER - T1CTRL

T CONTROL >0103

7	6	5	4	3	2	1	0	
CAPTURE LATCH VALUE								READ
START	SOURCE	0	PRESCALE LATCH VALUE					WRITE

TIMER 2 DATA REGISTER - T2DATA

P18 > 112

7	6	5	4	3	2	1	0	
CURRENT TIMER VALUE								READ
TIMER LATCH VALUE								WRITE

TIMER 2 CONTROL REGISTER - T2CTRL

P19 > 113

7	6	5	4	3	2	1	0	
0	0	0	0	0	0	0	0	READ
STRT	SRCE	CSCD	PRESCALE LATCH VALUE					WRITE

Bit 7 of the timer control register (TCNTL) is the start bit for the programmable timer. When a zero is written to the start bit, the timer chain is disabled or frozen at the current count value. When a one is written to the start bit (TCNTL (7)) of the TCNTL, (even if it already was one), the prescaler and counter decrementers are loaded with the corresponding latch values, and the timer/event counter operation begins.

RTC Mode

In the RTC mode, the internally-generated $f_{osc}/32$ (divide by 4 option) is the decrementer clock source. Each positive-edge transition of the $f_{osc}/32$ signal decrements the count chain. This mode allows a program to periodically call a service routine, such as a display refresh, by simply setting the prescale latch value and the timer latch value so the routine is called at the desired frequency.

EC Mode

When Timer 1 or Timer 2 is in the EC mode the counter functions the same as in the RTC mode except that pins A7 and A6 are the decrementer clock sources respectively. A positive-edge transition on these pins decrements the count chain. Note that this will allow INT2 and INT5 to function as a positive edge-triggered external interrupt by loading a start value of zero into both the prescaler and timer latches. A positive transition on A7 or A6 will decrement the corresponding timer through zero and generate an INT2 or INT5. This timer mode can also be used as an externally provided RTC if the external clock is input to I/O pin A7. The maximum clock frequency on A7 in the EC mode must not be greater than the $f_{osc}/32$, assuming divide by 4 clock option, and the mimimum pulse width must not be less than five f_{osc} cycles (1.25 machine cycles).

TIMER AND PRESCALED CLOCK

The timer clock, whether internal or external, is prescaled by a 5-bit modulo-N counter. The prescaling value is determined by the least significant five bits of the timer control register. The actual prescaling value is equal to the timer control latch value plus one. Thus, a value of >88 in the timer control latch would result in a $f_{osc}/160$ clock output from the prescaler, assuming a ÷ 4 clock option.

An INT2 interrupt (for Timer 1) or an INT5 interrupt (for Timer 2) is momentarily pulsed when the prescaler and counter value decrements past the zero value together. This sets the INT2 or INT5 flag flip-flop. The prescaler and counter are them immediately reloaded with the contents of the timer latch (TL) and prescaler latch (PL) then Timer 1 will start decrementing with the new TL and PL value. The TL is loaded through the timer 1 data register (T1DATA) and Timer 2 Data Register (T2DATA) is loaded into Timer 2. This value is write-only. When read, the timer data register contains the current value of the counter. The PL is loaded through the Timer 1 control register (T1CTRL) for Timer 1 and the Timer 2 control register (T2CTRL) loads into Timer 2. When read, the T2CTRL contains the Capture latch (CL) value and the T2CTRL contains all zeros.

Interrupt Pulses

The period of the INT2 interrupt pulses may be calculated by the following formula:

$$t_{INT} = t_{CLK} (PL + 1) (TL + 1)$$

Where:

t_{INT} = period of timer interrupts

t_{CLK} = $32f_{osc}$ (÷ by 4 option) for internal $\overline{RTC}$ mode or period of inut external EC mode

PL = Prescaler Latch value

TL = Timer Latch value

For example, with a 10MHZ external crystal, machine cycle frequency t_{CLK} is 32/10MHZ = 3.2 microseconds. Using the internal clock as a source if the Prescaler Latch (PL) is initialized to decimal 5 and the TL is initialized to decimal 99, the interval between timer interrupts t_{INT} is (3.2) (5+1) (99+1) = 1920 microseconds. Using the internal clock with a 10MHz crystal, the minimum timer interval (PL=0, TL=0) is 3.2 microseconds, and the maximum timer interval (PL=31, TL=255) is 26.21 milliseconds.

At the falling edge of the INT3 input, the Timer 1 value is loaded into the Capture Latch (CL). When read, the TCNTL contains the CL value. This feature provides the capability to determine when an external event occured relative to the internal timer.

TIMER 2

Timer 2 is similar to Timer 1 except the T2DATA register is P19, there is no Capture Latch associated with Timer 2, and INT5 is generated by Timer 2. In addition, T2CTRL also contains the CASCADE bit (5). This bit is used in conjunction with T2CTRL SOURCE bit (6) to determine the decrementing source of Timer 2.

A CASCADE bit of 1 selects the interrupt generated by Timer 1 (INT2) as the decrementing input to Timer 2. The CASCADE bit overrides the SOURCE bit, i.e., if the CASCADE bit is set to 1 the SOURCE bit has no effect.

As with Timer 1, a SOURCE bit of 0 selects the internally generated $f_{osc}/32$ signal (- by 4 option) and places the timer in the Real Time Clock (RTC) mode. A SOURCE bit of 1 selects the external clock source and places the Timer/Event Counter in the Event Counter (EC) mode.

The external EC input for Timer 2 is I/O bit (A6) which is also the serial clock (SCLK). SCLK is the I/O line (depending on mode of operation) for the baud rate generator used.

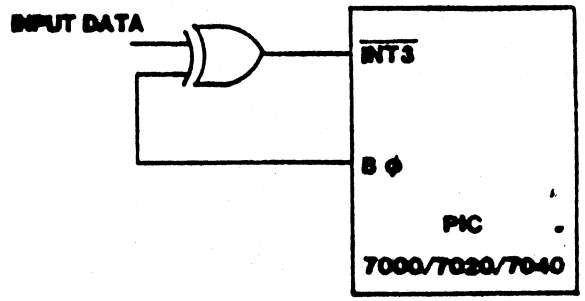
Driving the external EC line for Timer 2 with the SCLK produces the following modes for the I/O bit (A6):

a. With both SCLK and T2 external, the input signal drives the baud rate timer and Timer 2.
b. With SCLK external and T2 internal, the I/O bit (A6) drives the baud rate timer and f_{osc} 32/ (- 4 option) drives T2.
c. With SCLK and T2 internal, the SCLK pin is the 1 x baud rate signal and the T2 source is $f_{osc}/32$ (- 4 option).
d. With SCLK internal and T2 external, SCLK is the 1 x baud rate signal and drives T2. In this mode, the baud rate timer and Timer 2 are cascaded, with the baud rate timer driving Timer 2. This is done by setting the CASCADE bit to 0 and the Timer 2 SOURCE bit to 1. Timer 2 can then be cascaded with either Timer 1 or the baud rate timer.

Pulse Width Measurement

Through the use of the CL, the Timer/Event Counter can work with pulse width measurement applications. Only a simple exclusive-OR gate is needed to set up the PIC7001, or 7041 to handle a pulse-width modulated input, as shown below. In software, the user outputs the inverted input pulse train through one of the output lines (B0 in this case). This line is exclusive-ORed with the input data line, and the result is input to the INT3 pin. This causes the CL to be loaded with the current value of the timer at each transition of the input pulse train. The user's program can then compare these values to determine width values.

PULSE WIDTH MEASUREMENT



Pulse Width Modulation (PWM) Theory of Operation

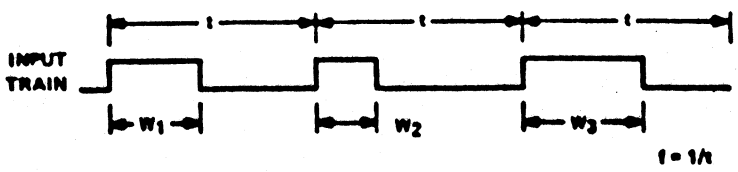
PWM involves the encoding of information in the width of pulses in a train of base frequency.

The figure below shows that the decoding of this information is equivalent to an (inverted) interval measurement. The following technique can be extended to interval measurement application:

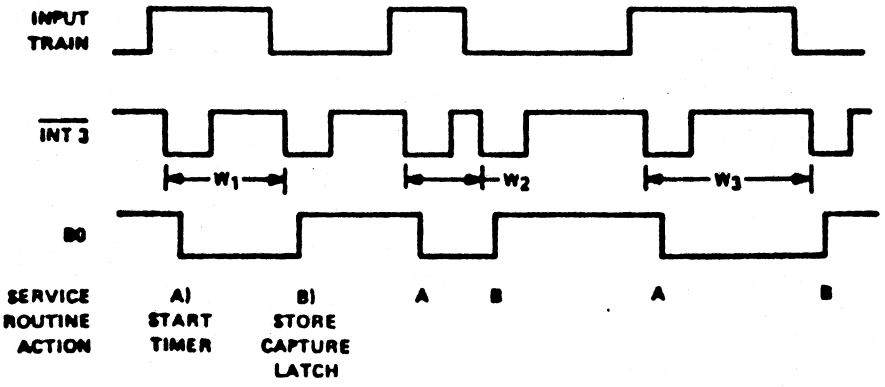
The 7041 is equipped to perform pulse measurement with the addition of a single exclusive OR-gate.

The edges of the PWM measurement are driven off of INT3 while the onboard counter times the event. The 7041 interrupt is structured so that the current value of the timer is captured at the CL (P3) on receipt of INT3. The actual time between events can then be derived from this captured value. The additional output B0 is used to disable INT3 between successive edges of input train.

PULSE WIDTH MODULATED PULSE TRAIN



7041 PWM INT3 TIMING



The decoded data, now encoded in the interval between INT3s, is available on alternate interrupts at the CL (P3). A sample INT3 service routine is:

```
INT3    XORP %>01,P6        TOGGLE B0
        DLC  R2             MARK YOUR PLACE
          ....
        BIJO %>01,R2,RSTRT  JUMP OFF OF MARKER
        MOVP P3,B           SAVE CAPTURE LATCH DATA
        RETI

RSTRT   MOVP %>80,P0        RESTART TIMER
        RETI
```

In this sample, R2 is used to keep the interval measurement on the proper portion on the pulse, and to flag the interrupt to the regular code. B0 saves the CL data for the regular code to interpret.

For long pulse widths, the prescale value can be adjusted to prevent the timer from rolling over before receiving an INT3. An alternate solution is to maintain a zero value of prescale, but use INT2 (the timer interrupt) to drive a software counter. A sample code is:

```
INT2    ORP%>80,P0          /CLEAR INT2 FLAG
          ....
        INC R4              /INCREMENT UPPER STAGE
        RETI                 COUNTER
```

NOTE: This sample code involves using the 7041 in a multi-interrupt environment. Care must be taken to ensure that a correct sequence of interrupts is performed.

MULTI-INTERRUPT PWM

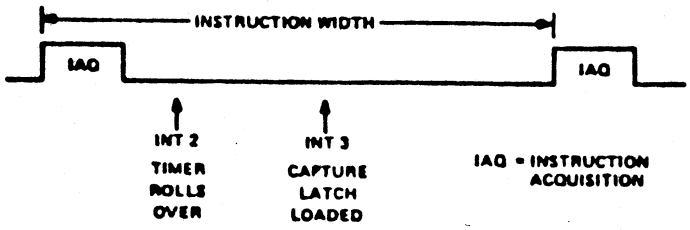
A simultaneous interrupt occurs when the INT3 service routine is delayed due to the receipt of a higher priority INT2 at the same time.

For example, when the user is operating the INT2 timer at high resolution (low value of prescale) to time intervals between successive INT3 events, the INT2 service routine increments a software controlled RAM byte. This byte serves as an upperstage byte for the timer, so the high resolution offered by a low value of prescale can be maintained.

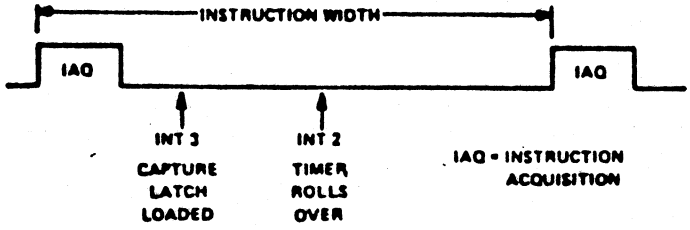
However, when both interrupts occur within an instruction cycle, one of the two sequences shown in the following figure has occured.

In the first sequence, if INT2 precedes INT3 within an instruction boundary, the receipt of INT2 implies that the timer has rolled over and its latch value (>FF) is reloaded into the current timer register. The current timer value was captured upon receipt of the interrupt (3). The INT2 service routine increments the software (RAM) counter and exits. The INT3 is then immediately serviced as the current timer value was captured upon receipt of the interrupt (3). The service routine reads the capture latch value, and a correct interval may be deduced from this capture value and the software upperstage counter value.

SIMULTANEOUS INTERRUPTS, INT2 PRECEDING



SIMULTANEOUS INTERRUPTS, INT3 PRECEDING



The second sequence that can occur is when INT3 precedes INT2 within an instruction boundary. As in the previous case, INT2 is serviced first. However, the current timer value is captured by hardware when INT3 comes in, before actual servicing of INT3. INT2 has not yet come in, and the hardware has therefore captured a timer value that has not rolled over. This timer value is likely to be near or at >00. The INT2 service routine, if it does not check for this condition (by testing the MSB of the timer for rollover) will increment the upper stage of software by default and will cause an incorrect value to be assumed for the interval. This condition happens because the timer (implemented in hardware) and the upperstage counter (software driven) are out of sync.

The following code will correct the situation:

```
INT     BTJZP   %>20,P0,OKAY    CHECK FOR PENDING INT3
        BTJO    %>80,P3,OKAY    CHECK CAPTURED VALUE
        JMP     RET3
OKAY    INC                     OKAY TO INCREMENT
                                UPPER STAGE
        ....
        RET3 RETI
```

INT3 then becomes:

```
INT3    XORP    %>01,P6         TOGGLE B0
        DEC     R2              MARK YOUR PLACE
        ....
        BJTO    %>01,R2,RSTRT   JUMP OFF OF MARKER
        MOVP    P3,B            SAVE CAPTURE LATCH DATA
        RETI

RSTRT   MOVP    %>80,P3         RESTART TIMER
        CLR     R4              RESTART UPPER STAGE
        RETI
```

I/O CONTROL REGISTERS

The I/O control registers are mapped into locations P0 (IOCNTL0) and P16 (IOCNTL1) of the Peripheral File as shown below. The memory expansion modes and individual interrupt masks and resets are controlled through these registers. The interrupt sources may also be individually tested by reading the interrupt flags. The interrupt flag values are independent of the interrupt enable values.

Writing a 1 to an interrupt cleat bit in an I/O control register will clear the corresponding interrupt flag. The interrupt flag is unaffected if a 0 is written to the interrupt clear bit.

Due to the read/modify nature of the bit manipulation instructions (ANDP, ORP, and XORP), it is possible that a pulsed interrupt could occur during the operation of these instructions on the IONCILO and IOCNIOL1 and be missed. These instructions could also cause the other interrupt flags to be unintentionally cleared or not cleared. For example, there is no problem if an XORP instruction is used to enable INT1 and not alter the condition of the INI1 flag (XORP %, >03, P0), as long as the flag flip-flop does not change state during instruction execution. However, if a short INI1 pulse occurs during execution a 0 may be read and a 1 would be written to reclear the INI1 flag. In this case, the INI1 pulse would be undetected by the processor. The same instruction would also affect the INI2 and INI3 flags as they are also located in IOCNILO. To avoid these occurrences, use the MOVP and STA instructions when writing data to IOCNIL0 and IOCNIL1.

I/O CONTROL REGISTERS

IOCNTLO

	7	6	5	4	3	2	1	0	
	MEMORY MODE 1	MEMORY MODE 0	INT3 FLAG	INT3 ENABLE	INT2 FLAG	INT2 ENABLE	INT1 FLAG	INT1 ENABLE	RLAD
P0									
	MEMORY MODE 1	MEMORY MODE 0	INT3 CLEAR	INT3 ENABLE	INT2 CLEAR	INT2 ENABLE	INT1 CLEAR	INT1 ENABLE	WRITE

IOCNIL1

	7	6	5	4	3	2	1	0	
	0	0	0	0	INT5 FLAG	INT5 ENB	INT4 FLAG	INT4 ENB	READ
P16									
	0	0	0	0	INT5. CLEAR	INT5 ENB	INT4 CLEAR	INT4 ENB	WRITE

The following code segment is an example of how the user can regulate the memory mode bits and individual interrupt masks and resets through program control:

```
IOCNTL  EQU P0

        MOVP    >3B,IOCNTL  SINGLE-CHIP MODE, CLEAR
                            ALL INT FLAGS ENABLE
                            INT1 AND INT3
        BTJOP   >08,IOCNTL  TEST IF INT2 SET
        ANDP    >E5,IOCNTL  CLEAR AND DISABLE INT3

LABLL   EQU$
```

NOTE: This example is one of the few situations where use of the ANDP instruction is possible.

On $\overline{\text{RESET}}$, the IOCNILO register is written with all 0's. This disables INT1, INT2, and INT3 individually. The IOCNTL1 register is not written to during $\overline{\text{RESET}}$. In order to ensure that INT4 and INT5 are also individually disabled, it is recommended that all 0's be written to IOCNTL1 register immediately after $\overline{\text{RESET}}$. Note that following $\overline{\text{RESET}}$, all interrupts are globally disabled because the I bit in the status register is reset to 0.

Because the state of the interrupt flag flip-flops are undetermined, it is recommended that the flags be cleared by writing a 1 to bit positions 1, 3, and 5 in P0 and positions 1 and 3 in P16. Bits 7 through 4 in P16 will stay 0 and cannot have a 1 written to them. The recommended RESET code sequence is as follows:

```
RESET  MOVP>3F,P0      Clear INT Flags 1,2, and 3
                       and enable INTs 1,2, and 3 and
                       place in Single-Chip mode.
       MOVP>00,P16     Clear INT Flags 4 and 5 and
                       disable INT4.
       MOVP value,P8   Load initial Port C outputs.
       MOVP mask,P9    Load Port C data direction.
       MOVP value,P10  Load initial Port D outputs.
       MOVP mask, P11  Load Port D data direction.
       MOVP mask, P2   Load TL value
       MOVP>80 P3      Start timer, timer source,
                       internal prescaller latch.
```

SERIAL PORT

Description

The PIC7001/7041 contains a serial port which greatly enhances its I/O and communication capability. The serial port can operate in several modes which let the 7001/7041 interface with Universal Asynchronous Receiver/Transmitter (UART) peripheral devices, as well as multiple microcomputers (7041, MC6801, i8051). These serial links are implemented using standard asynchronous protocols. These multiprocessor protocols, are compatible with those used by tthe Motorola MC6801 and Intel i8051.

A second mode, isosynchronous, permits very high transmission rates.* The serial port also has a serial I/O mode which can be used to expand I/O lines using external shift registers, and to communicate with peripheral devices requiring a non-UART serial input (e.g. display drivers).

An on-chip hardware serial port saves ROM code and allows much higher transmission rates than could be achieved in software. The full-duplex serial port has a double buffered transmitter and receiver.

The serial port consists of a receiver (RX), transmitter (TX) and Timer 3. The complete functional definition of the serial port is programmed by the 7041 software. A set of control words must first be sent out to the serial port to initialize it, so that it supports the desired communications format. These control words will determine the baud rate, character length, even/odd/off parity, number of stop bits, etc.

The serial port is controlled and accessed through registers in the peripheral file. The registers associated with the serial port are:

REGISTER	NAME	TYPE	FUNCTION
P17	SMODE	WRITE	Serial Port Mode
	SCTL0	WRITE	Serial Port Control-0
	SSTAT	READ	Serial Port Status
P20	T3DATA	R/W	Timer 3 Data
P21	SCTL1	R/W	Serial Port Control-1
P22	RXBUF	READ	Receiver Buffer
P23	TXBUF	WRITE	Transmission Buffer

The SMODE register is the receive/transmit (RX/TX) write-only mode and format register. The SCTL0 and SSTAT are the RX/TX write-only control register and read-only status register, respectively. All of these registers are accessed from P17. The first write after a hardware or serial port reset accesses SMODE. All subsequent writes access SCTL0. These registers are common to both RX and TX, both of which will have the same mode and frame format.

The T3DATA register accesses the 8-bit timer of Timer 3, and is similar to T1DATA and T2DATA. The SCTL1 register is a read/write control register for the RX/TX and Timer 3.

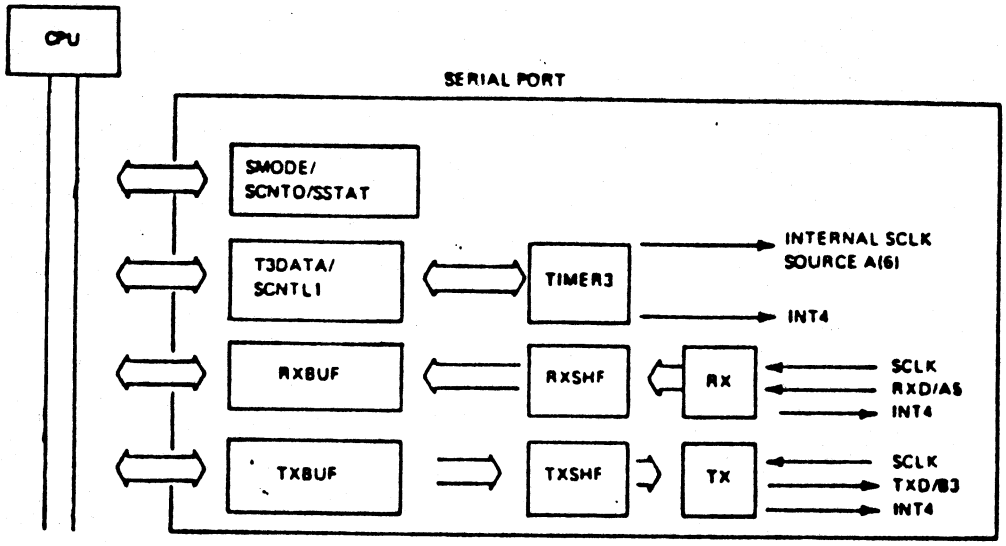
The RXBUF is a read-only register containing data from the RX. The RXBUF is double buffered with the internal shift register (RXSHF) so that the PIC7041 CPU has at least a full frame to read the received data before the RX may overwrite it with new data.

*Isosynchronous is the term given to this second communication mode of the serial mode. This mode has the same frame format as the asynchronous mode, but uses only one serial clock (SCLK) cycle per data bit as opposed to 16 SCLKs per data bit for the asynchronous mode. This allows transmission rates 16 time those of the asynchronous mode.

The TXBUF is a write-only register from which the TX takes the data it transmits. It is double buffered with the TX shift register (TXSHF), so that the 7041 CPU has a full frame to write new data before TXBUF becomes empty.

The TXD and TXD lines use I/O lines B3 and A5 respectively. This configuration allows the TXD and RXD pins to be used as I/O pins if desired. If serial port transmission is disabled, then TXD follows B3. If reception is disabled, then no receiver interrupts occur and A5 is an input bit.

SERIAL PORT FUNCTIONAL BLOCKS

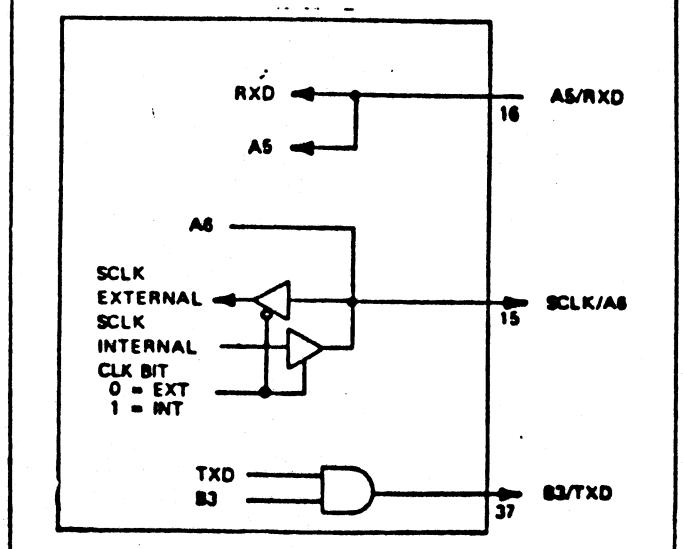


Clock Sources And Serial Port Modes

The serial port can be driven by an internal (Timer 3) or external baud rate generator. The source of the serial clock (SCLK) is determined by the clock (CLK) bit, SCIL1(6). An external clock source is input on the high impedance A(6) line. An internal clock source is output on the low impedance A(6) line, being derived from Timer 3 via a ÷ 2 clock (f_{osc}/16 for ÷ 2 option, Ø/8) as shown below. The internally generated SCLK has an equal mark-space ratio. The current value of SCLK (internal or external) can be determined by reading A(6). The RX receives data on the rising SCLK edges and the TX transmits data on the falling SCLK edges.

The RX/TX has three communication modes; asynchronous devices (capable of UART communication), isosynchronous, and serial I/O. The serial I/O mode is used to link the serial port to shift registers for simple I/O expansion. The isosynchronous and asynchronous communication modes are used to link devices with UART communication capability, and have extra features for two formats of multiprocessor communication. In all modes I/O is NRZ format, data value 1 = high level, and data value 0 = low level.

SERIAL PORT I/O LOGIC

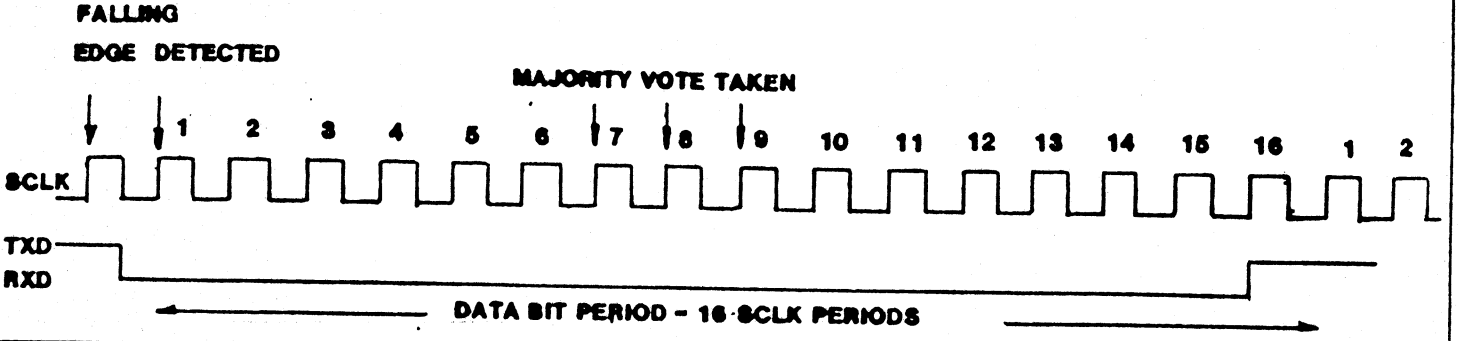


Asynchronous Communication Mode

When the serial port is operating in the asynchronous communication mode, the frame format consists of a start bit, five to eight data bits, even/odd/-no parity, and one or two stop bits. The bit period is 16 times the SCLK period.

RX operation is initiated by reception of a valid start bit, which consists of a negative edge (1 and then 0 in adjacent SCLK periods) followed by a majority value of 0, occurring seven, eight, and nine SCLK periods after the negative edge. This sequence provides false start bit rejection and also locates the center of bits in the frame, where the bits will be read on a majority basis. The figure below illustrates the asynchronous communication format, with a start bit showing how edges are found and majority vote taken.

ASYNCHRONOUS COMMUNICATION FORMAT

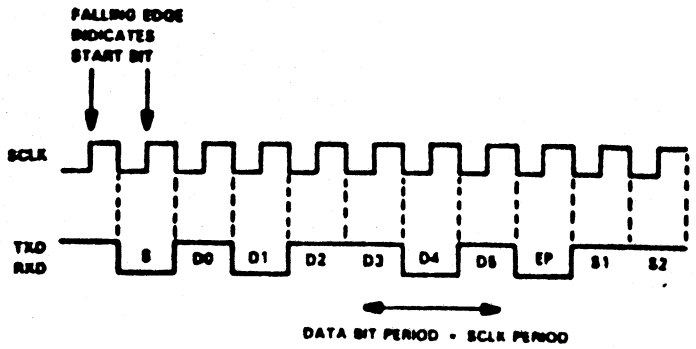


Since the RX synchronizes itself to frames, the external transmitting and receiving devices do not have to use the same SCLK; it may be generated locally. If the internal SCLK is used it will be output continuously on A(6).

ISOSYNCHRONOUS COMMUNICATION MODE

In this mode, the frame format consists of a start bit, five to eight data bits, even/odd/no parity, and one or two stop bits. The bit period equals the SCLK period. RX operation is initiated by reception of a valid start bit, which consists of a negative edge. Bits are read on a single value basis. Since the RX does not synchronize itself to data bits the transmitter and receiver must be supplied with a common SCLK. If the internal SCLK is used it is output continuously on A(6). The figure below illustrates the isosynchronous communication format, with a complete frame consisting of a start bit, six data bits, even parity, and two stop bits.

ISOSYNCHRONOUS COMMUNICATION FORMAT



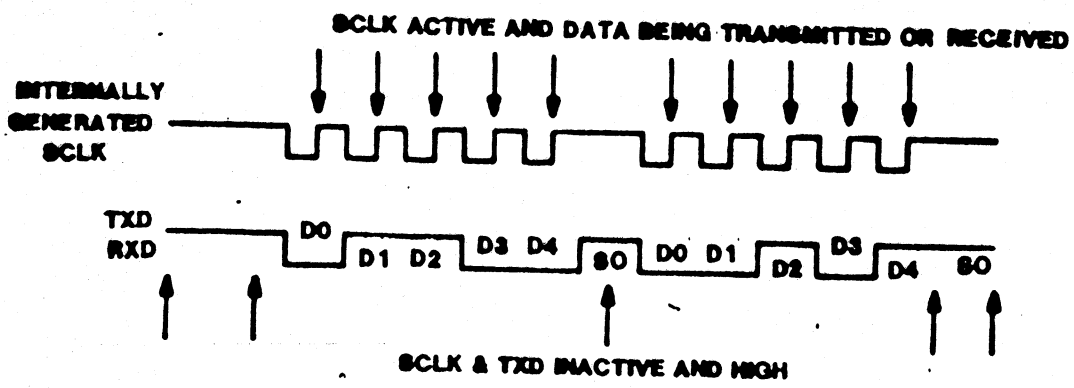
In both the asynchronous and isosynchronous communication modes, when a frame is fully received, RXBUF is loaded from RXSHF, RXRDY and INT4 FLG are set to 1, and the error status bits are set accordingly. RXRDY is reset to 0 when the CPU reads RXBUF.

Transmission is initiated after the CPU writes to TXBUF. This sets TXRDY to 0. Once TXSHF is empty, it is loaded from TXBUF, setting TXRDY and INT4 FLG to 1. Upon completion of the transmission, TXSHF will reload if TXBUF is full; if not the TX will idle and TXL will be 1 until TXBUF is written to.

Serial I/O Communication Mode

When the serial I/O mode is in operation, the frame format is five to eight data bits and one stop bit, with no corresponding clock edge for the stop bit. The clock is off for the stop bit. The bit period is equal to the SCLK period. TX operation is initiated by writing to TXBUF, when TXRDY equals 1. RX operation is initiated by writing a 1 to the RXEN bit. The figure below illustrates the serial I/O format for two back to back frames, each containing five data bits.

SERIAL I/O COMMUNICATION FORMAT



An internal SCLK source will be output on A(6). In the serial I/O mode, SCLK will only be active when data is being transmitted or received, otherwise A(6) will have a one value. An external SCLK should not be selected when in the serial I/O mode; this operation is undefined.

Multiprocessor Communication

When the serial port is in either the asynchronous or isosynchronous communications mode, the multiprocessor communication formats are available. These formats are used to transfer information between many microcomputers on the same serial link. Information is transferred as a block of frames from a particular source to some destinations(s). The 7001/7041 has features to identify the start of blocks, and suppress interrupts and status information from the RX until a block start is identified.

In both multiprocessor modes the sequence is as follows: The serial port wakes up at the start of a block and the 7001/7041 CPU reads the first few frames (containing a destination address). If the block is addressed to the microcomputer the CPU proceeds to read the rest of the block; if not it puts the serial port to sleep again and therefore will not receive serial port interrupts until the next block start.

In order to provide more flexibility, the 7001/7041 implements two multiprocessor protocols, one supported by Motorola and the other by Intel. These protocols are described in the following paragraphs. The Motorola protocol is compatible with the Motorola MC6801 processor mode; the Intel protocol with the Intel protocol Intel i8051. The mode of 7001/7041 multiprocessor protocol is software selectable via the MULTI bit in the SMODE register. Both formats use the wake up (WU) and SLEEP flags to control the TX and RX features of these modes.

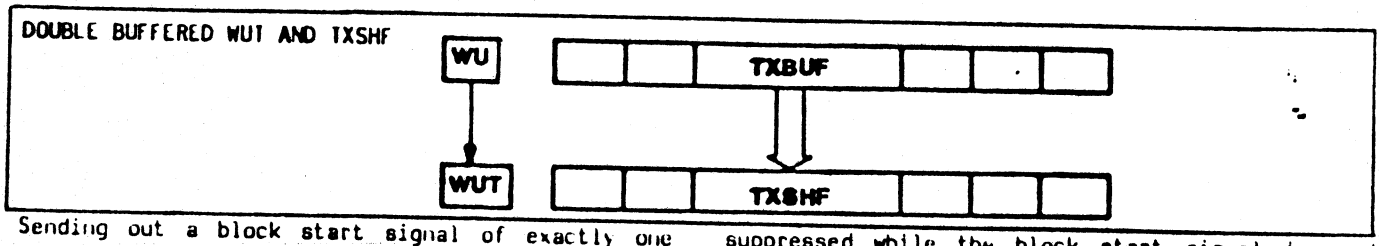
Motorola (MC6801) Protocol

In this protocol, blocks are distinguished by having a longer idle time between the blocks than between frames in the blocks. An idle time 10 bits or more after a frame indicates the start of a new block.

In the Motorola mode of multiprocessor communications, the processor wakes up (serial port resets the SLEEP bit to 0) after the block start signal. The processor will now recognize the next serial port interrupt. The CPU then receives the address sent out by the transmitter and compares this address to its own. If the CPU is addressed, the 7001/7041 will not set the SLEEP bit and receive the rest of the block. If the CPU is not addressed, it sets the SLEEP bit (in software) to a 1. This lets the CPU continue to execute its main program without being interrupted by the serial port. The serial port will set the SLEEP bit to 0 whenever it detects a block start signal.

There are two ways to send a block start signal. The first is to deliberately leave an idle time of 10 bits or more by delaying the time between the transmission of the last frame of data in the previous block and the address frame of the new block. In the second way, the 7001/7041 implements a more efficient method of sending a block start signal. Using the wake up WU bit, an idle time of exactly one frame (timed by the serial port) can be sent. The serial communications line is therefore not idle any longer than necessary.

Associated with the WU bit is the wake up temporary (WUT) flag. WUT is an internal flag, double buffered with WU. When TXSHF is loaded from TXBUF, WUT is loaded from WU and WU is reset to 0.



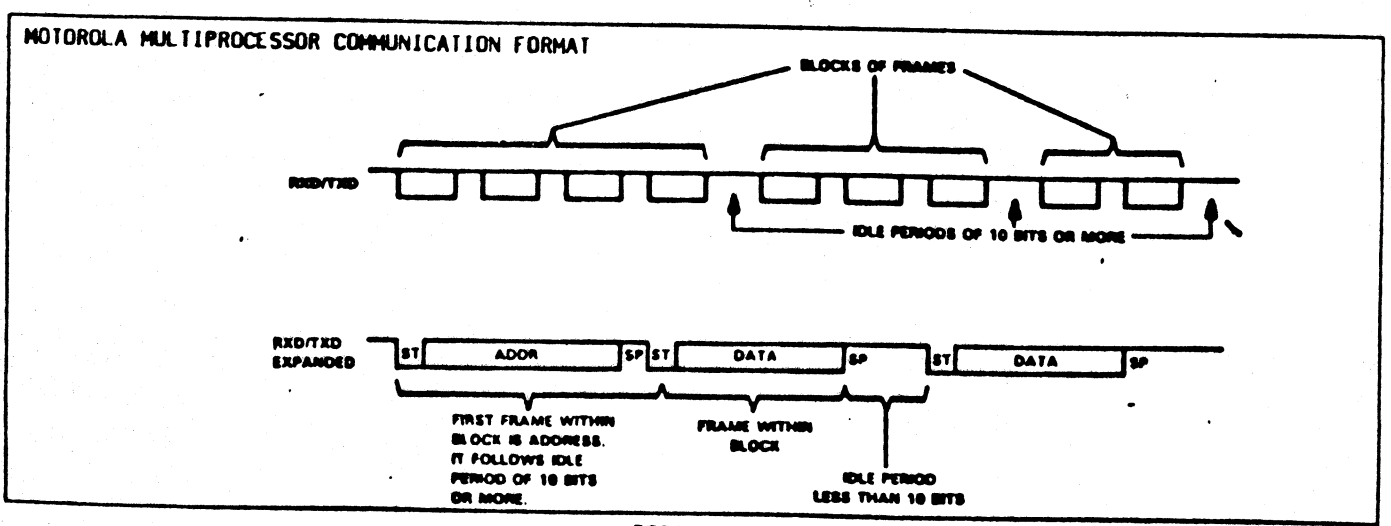
Sending out a block start signal of exactly one frame time is accomplished as follows:

A 1 must first be written to the WU bit. Then a data word (don't care) must be written to the TXBUF. When the TXSHF is free again, the contents of the TXBUF are shifted to the TXSHF, and the WU value is shifted to WUT. If the WU bit had been set to a 1, the start, data, and parity bits will be suppressed and an idle period of one frame, timed by the serial port, will be transmitted.

The next data word, shifted out of the serial port after the block start signal, will be the second data word written to the TXBUF after writing a 1 to the WU bit. The first data word written is suppressed while the block start signal is sent out, and ignored after that.

Writing the first don't care data word to the TXBUF is necessary so the WU bit value can be shifted to WUT. After the don't care data word is shifted to the TXSHF, the TXBUF (and WU if necessary) may be written to again, since WUT and TXSHF are both double buffered.

Although the RX still operates then the SLEEP bit is 1, it will not set RXRDY, INT4 FLG, or the error status bits to 1. The RX will set the SLEEP bit to 0 if it times an appropriate 10 bit idle time on RXD. The Motorola multiprocessor communication format is shown below.



Intel (i8051) Protocol

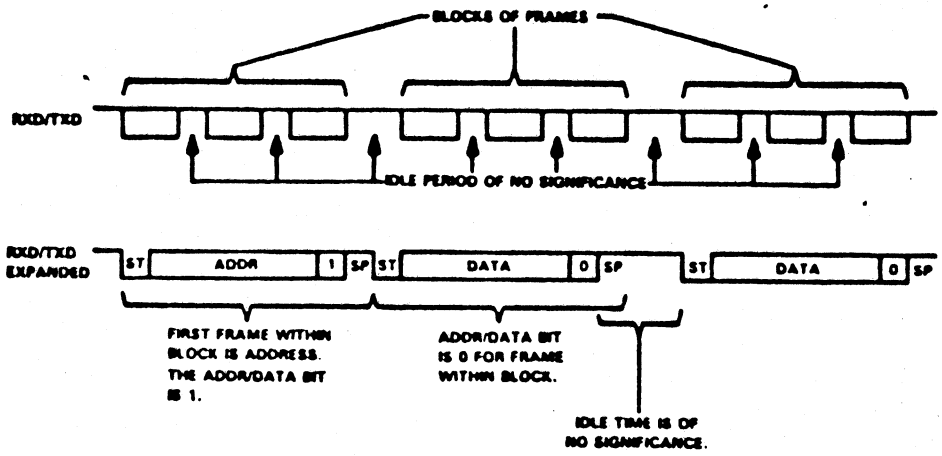
In the Intel protocol, the frame has an extra or address bit just before the parity bit. Blocks are distinguished by the first frame(s) in the block with the address bit set to 1, and all other frames with the address bit set to 0. The idle period timing is irrelevant.

The WU bit is used to set the address bit. In the TX, when the TXBUF and WU are loaded into the TXSHF and WUT, WU is reset to 0 and WUT is the value of the address bit of the current frame. Thus, to send an address, the WU bit must be set to a 1, and the appropriate address value should then be written to the TXBUF. When this address value is transferred to the TXSHF and shifted out, its address bit will be sent as a 1, which flags the other processors on the serial link to read the address. Since the TXSHF and WUT are both double buffered, the TXBUF and WU may be written to immediately after TXSHF and WUT are loaded. To transmit non-address frames in the block, the WU bit must be left at 0.

On the serial link, all processors set their SLEEP bit to 1. In this way they will only be interrupted when the address bit is a 1. When the processors receive the address of the current block, they compare it to their own addresses. Those processors which are addressed set their SLEEP bit to a 1, so they will read the rest of the block.

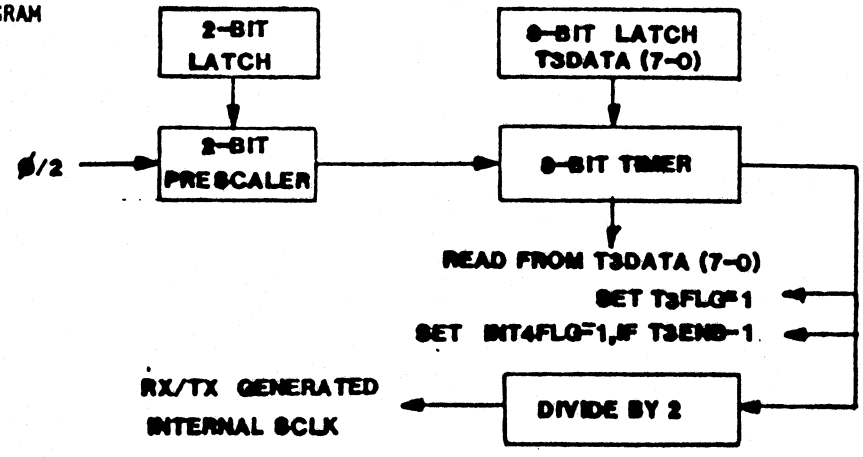
Though the RX still operates when SLEEP bit is 1, it will not set RXRDY, INT4 FLG, or the error status bits to 1 unless the address bit in the received frame is a 1. The RX does not alter the SLEEP bit; this must be done in software. The Intel multiprocessor communication format is shown below.

INTEL MULTIPROCESSOR COMMUNICATION FORMAT



Timer 3

TIMER 3 BLOCK DIAGRAM



Timer 3 is accessed through T3DATA (similar to T1DATA and T2DATA), and SCTL1 (shared with RX/TX functions). The clock source for Timer 3 is internal only, and has a frequency of Ø/2. Timer 3 is a free running clock and is updated with new timer values when it decrements through zero.

Timer 3 consists of a 2-bit prescaler and an 8-bit timer. Both the prescaler and the timer are reloaded from 2-bit and 8-bit latches respectively when they decrement through zero. The latches are write only, but the 8-bit timer can be read via T3DATA.

The Timer 3 output goes to the serial port via a divide by two circuit, producing an equal markspace ratio internal SCLK. The baud rate generated by Timer 3 is user programmable and is determined by the value of the 2-bit prescaler and the 8-bit timer latch. The equations for determining the baud rates for both the asynchronous and isosynchronous modes are as follows:

$$\text{Asynchronous Baud Rate} = \frac{Ø}{64*(P+1)*(L+1)}$$

$$\text{Isosynchronous Baud Rate} = \frac{Ø}{4*(P+1)*(L+1)}$$

where:

Ø = frequency of internal machine clock
P = Timer 3 prescaler value
L = Timer 3 latch value

For example, to program the serial port to operate at 300 baud in the asynchronous mode (with Ø = 2.5 MHz), the prescaler value is set to 0 and the latch value set to 129 decimal, or >81.

The Timer 3 output always sets T3FLG to 1, and sets INT4 FLG to 1 if T3ENB is a 1 when the timer and prescaler decrement through 0. This allows Timer 3 to be used as a utility timer if it is not used by the serial port. Timer 3 and its flags are not affected by the serial port software reset, UR. Therefore, Timer 3 may be used independently of the serial port.

Serial Port Register
Mode Register (SMODE)

SMODE is a write-only register and is accessed through P17 in the Peripheral File. It describes the character format and type of communications mode (asynchronous or isosynchronous). SMODE is only accessible after a reset operation (Hardware or software), and must be the first register written to in the serial port immediately following a reset. After writing to the SMODE register, it cannot be accessed without first performing a reset operation. The first operation to location P17 in the Peripheral File, immediately following a reset, will access the SMODE register. All subsequent writes to P17 will access the control register (SCTL0).

SMODE

7	6	5	4	3	2	1	0
SIOP	SIO	PLVLN	PEN	CHAR1	CHAR0	COMM	MULTI

0 = Motorola protocol
1 = Intel protocol

0 = Isosync communication
1 = Async communication

00 = 5 bits/character
01 = 6 bits/character
10 = 7 bits/character
11 = 8 bits/character

1 = Parity enabled
0 = Parity disabled

1 = Even parity
0 = Odd parity

0 = Serial I/O mode
1 = Communication Mode

0 = One stop bit
1 = Two stop bits

NOTE: If the serial port is configured so that some features are irrelevant, then the corresponding flags are don't care. For example, when configured in the serial I/O mode, bits 7, 4, 1, and 0 are don't cares.

Multiprocessor Mode (MULTI)

There are two possible multiprocessor protocols, the Motorola and the Intel. Setting this bit to a 0 selects the Motorola protocol; setting it to a 1 selects the Intel protocol.

Communications Mode (COMM)

This bit determines the serial port mode of communication. Setting the bit to 1 selects the asynchronous mode. In this mode the bit period is 16 times the SCLK period and bits are read on a two out of three vote basis. Setting the bit to 0 selects the isosynchronous mode, the bit period is equal to the SCLK period and bits are read on a single value basis.

Number of Bits per Character (CHAR0/1)

Characters are programmable to 5, 6, 7 or 8 bits. Characters of less than 8 bits are right-justified in RXBUF and TXBUF and are padded with leading blanks in RXBUF, but not in TXBUF.

Parity Enable (PEN)

If parity is disabled then no parity bit is generated during transmission or expected during reception. A received parity bit is not transferred to the RXBUF with the received data as it is not considered one of the data bits when programming the character field.

Parity Even (PENEVN)

If PEN is set, then this bit defines odd or even parity according to the number of odd or even 1 bits in both transmitted and received characters.

Serial I/O or Communication Mode (SIO)

This bit determines whether the serial port operates in the serial I/O mode or one of the communication modes. Setting this bit to a 0 sets the serial port in the serial I/O mode. Deletion of the start and stop bits, in conjunction with an internal 1x clock, allows ease of I/O expansion by use of external shift registers. Setting this bit to a 1 selects the communication mode. When this bit is set to 1 the COMM bit determines whether the serial port is in the asynchronous or isosynchronous mode.

Number of Stop Bits (STOP)

This bit determines the number of stop bits sent when the serial port is in one of the communication modes. Setting this bit to a 0 selects one stop bit, and setting it to a 1 selects two stop bits. The receiver checks for one stop bit only.

Serial Control 0 Register (SCTL0)

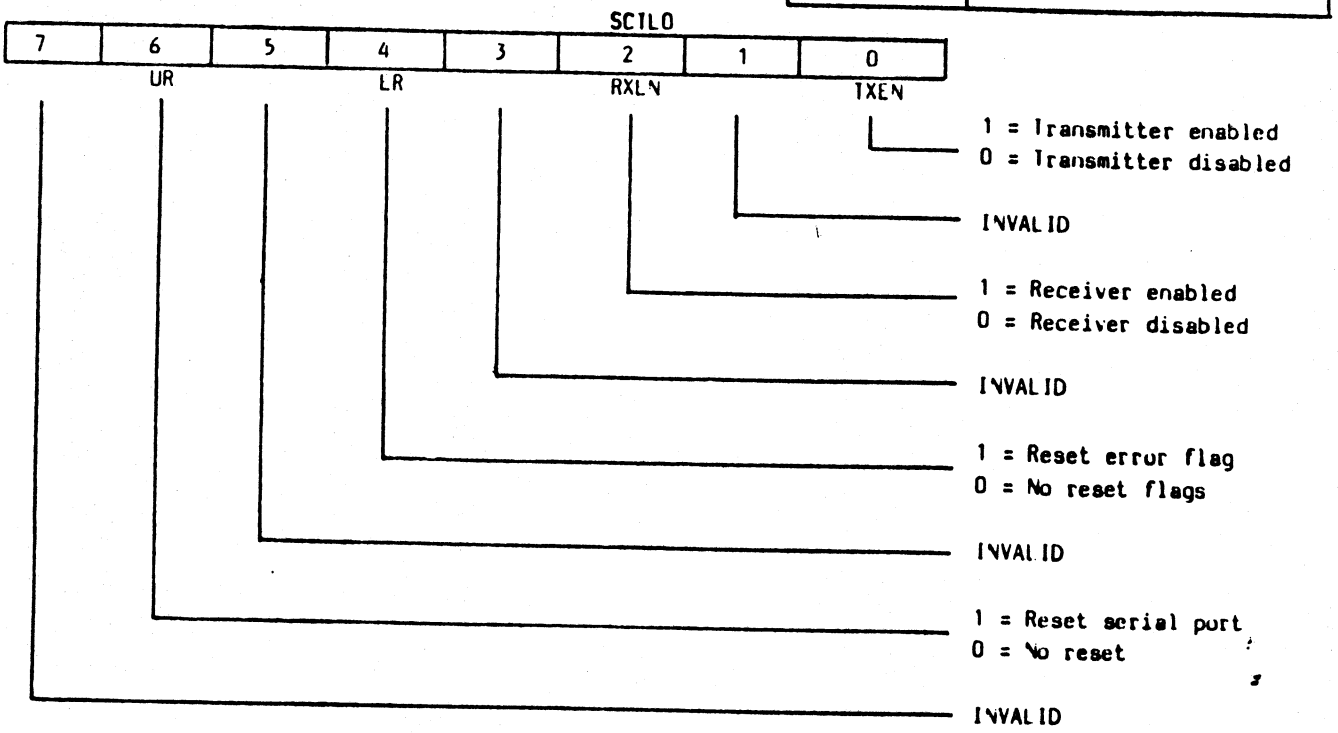
SCTL0 is a write-only register, and is accessed through P17 of the Peripheral File. The SCTL0 register is used to control the serial port functions, such as transmit and receive enable, clearing of error flags and software reset. After a hardware or software reset, the SMODE register must be written to before accessing the SCTL0 register, since the SMODE and SCTL0 registers are accessed through the same location. Any subsequent writes to this register location (P17) will load the SCTL0 register. SCTL0 is cleared by a reset (hardware or software).

Transmit Enable (TXEN)

Data transmission through TXD cannot take place unless this bit is set to a 1. When resetting to a 0, the transmission is not halted until all the data previously written to TXBUF has been sent. TXEN is set to 0 by a reset (hardware or software).

Receive Enable (RXEN)

In the communication modes (asynchronous and isosynchronous) setting the RXEN bit to 1 allows RX to set INT4 FLG, and enable RXRDY. When reset to 0, this bit prevents received characters from being transferred into the receiver buffer, and no RXRDY interrupt is generated. However, the receiver shift register continues to assemble characters. Thus, if RXEN is set partially through reception of a character, it will be transferred complete into RXBUF. In the serial I/O mode writing a 1 to RXEN initiates RX operation. If an internally generated SCLK is used, a gated SCLK at A(6) is enabled. When the entire frame is received, RX disables SCLK and sets RXRDY and INT4 FLG to a 1, and RXEN to 0. RXEN has no direct effect on RXRDY or INT4 FLG in this mode. RXEN is set to 0 by UR.



Error Reset (ER)

Writing a 1 to this bit clears all three error flags in the SSIAT register (PE, OE, FE). Writing a 0 has no effect.

Software Reset (UR)

Writing a 1 to this bit puts the serial port in the reset condition, and enables the SMODE register for initialization. A(6) is put in the high impedance state (input), the IXD signal is held at 1, so that the B(3) line may be used as a general purpose output line. Until a 0 is written to UR, all affected logic is held in the reset state. UR must be set to 0 before the 7001/7041 CPU can write a 1 to CLK and output SCLK on Port A. UR is set to 1 by reset (hardware).

Serial Port Status Register (SSIAT)

The Status Rebister is a read-only register and is accessed through P17 of the Peripheral File. It is used for determining the status of the serial port. Bits, 0, 1, and 6 of this register are cleared by a reset (hardware or software).

Transmitter Ready (TXRDY)

The TXRDY bit is set by the transmitter to indicate that TXBUF is ready to receive another character, and is automatically reset when a character is loaded. The serial port interrupt (INT4) is issued at the same time (if enabled) the TXRDY bit is set. This bit is set to 1 by UR.

Receiver Ready (RXRDY)

The RXRDY bit is set by the receiver to indicate that RXBUF is ready with a new character, and is automatically reset when the character is read out. The serial port interrupt (if enabled) is issued at the same time the RXRDY bit is set. RXRDY is set to 0 by UR.

Transmitter Empty (TXE)

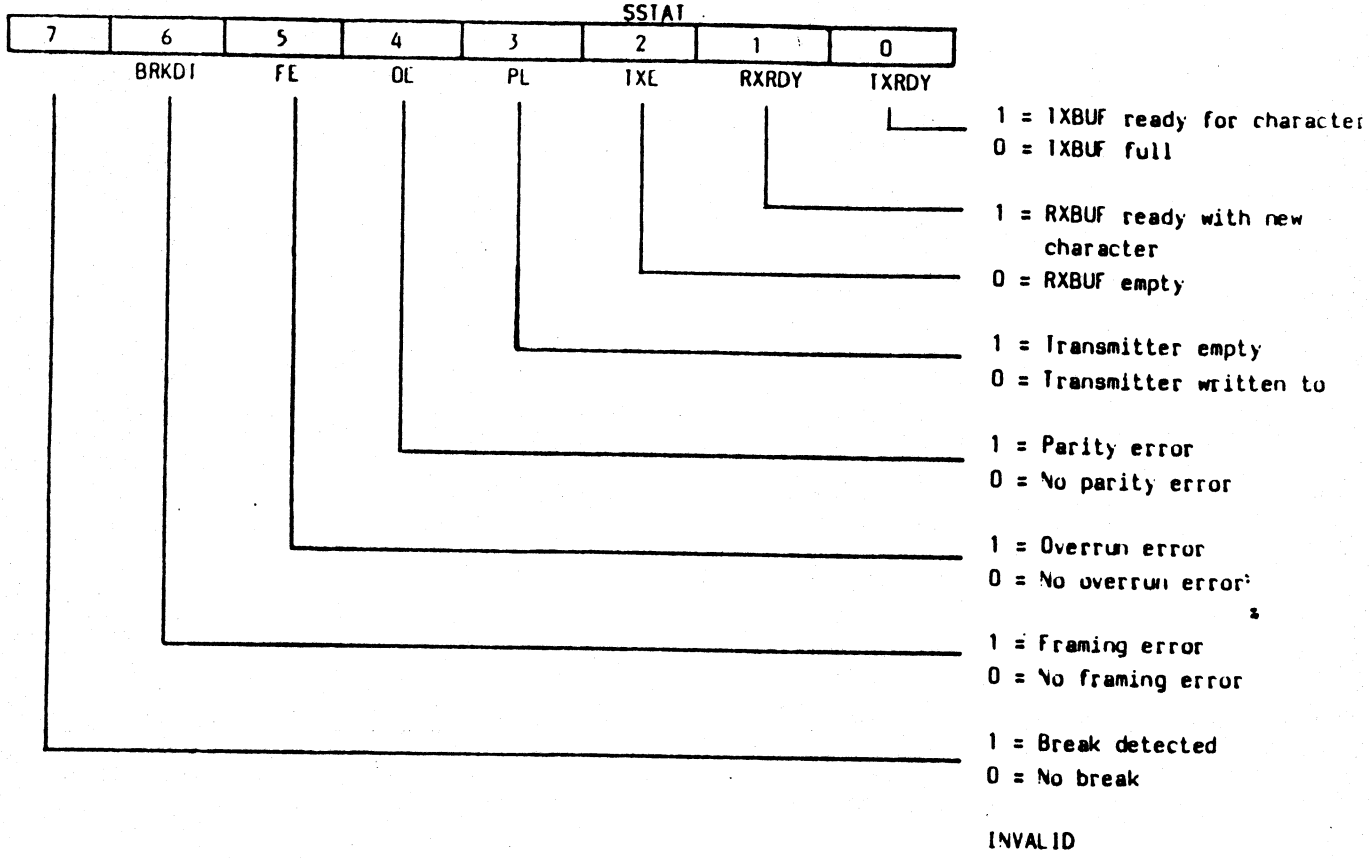
The TXE is set to 1 when the transmitter shift register and TXBUF are empty, and reset to 0 when the TXBUF is written to. This bit is set to 1 by UR.

Parity Error (PE)

PE is set when a character is received with a mismatch between the number of 1s and its parity bit. This bit is reset by the ER bit in SCTLO.

Framing Error (FE)

The FE bit is set when a character is received with a 0 stop bit. The stop bit indicates that synchronization with the start bit has been lost and the character is incorrectly framed. FE is reset by the ER bit in SCTLO.



Overrun Error (OE)

The OE bit is set when a character is transferred into RXBUF before the previous character has been read out. The previous character is overwritten and lost. OE is reset by the ER bit in SCIL0.

Break Detect (BRKDT)

The BRKDT bit will show that a break condition has occurred. BRKDT is set if the RXD line remains continuously low for 10 bits or more, starting from the end of a frame (stop bit). When the break ends, BRKDT is set to a 0 immediately. In the serial I/O BRKDT remains a 0. This bit is reset to 0 by UR. A break is generated by setting Port B bit 3 low. Setting Port B bit 3 high again resumes operation of the TXD line.

The Serial Port I/O Logic diagram shows how the TXD and RXD lines are multiplexed on I/O lines B3 and A5 respectively. This configuration allows the TXD and RXD pins to be used as I/O pins if desired. If transmission is disabled, then TXD follows B3. If reception is disabled, then no receiver interrupts occur and A5 is an input bit.

Serial Control 1 Register (SCIL1)

The SCIL1 is a read/write register and is accessed through P21 in the Peripheral File. This register is used to control the source of SCLK, multiprocessor communications, Timer 3 interrupt, and the Timer 3 prescaler value.

Baud Rate Clock (PRE3 (1, 0))

The internal clock input to the Timer 3 is either f_{osc}/16,/32,/64, or/128 (- 2 option) and f_{osc}/32,/64,/128, or /256 (- 4 option). These bits act as the prescale bits for Timer 3 and have no effect if an external baud rate clock source is selected through the CLK bit.

Timer 3 Interrupt Enable (T3ENB)

When T3ENB is set to a 1, Timer 3 will set INT4FLG to a 1 when it sets T3FLG to a 1. T3ENB is reset to 0 by a hardware reset, but not by UR. This allows Timer 3 to operate independently of the serial port.

SCTL1

7	6	5	4	3	2	1	0
	CLK	SLEEP	WU	T3FLG	T3ENB	PRE3(1)	PRE3(0)

- PRE3(1), PRE3(0): 2-Bit prescaler for Timer 3
- T3ENB: 0 = Disables T3 interrupt; 1 = Enables T3 interrupt
- T3FLG: 0 = Resets T3FLG; 1 = Set by T3
- WU: Controls TX multiprocessor communication
- SLEEP: Controls RX multiprocessor communication
- CLK: 0 = External clock; 1 = Internal clock
- 7: INVALID

Timer 3 Interrupt Flag (T3FLG)

The T3FLG bit is set to a 1 when both the Timer 3 prescaler and Timer 3 decrement through zero together. T3FLG indicates that Timer 3 was the source of the serial port interrupt. T3FLG must be cleared by software in the T3 interrupt service routine, since it is not cleared when the INT4 vector is fetched by the CPU. This bit is also reset to 0 by a hardware reset, but not by UR. This allows Timer 3 to operate independently of the serial port.

Wake Up (WU)

The WU bit controls the TX features of the multiprocessor modes. WU is reset to 0 by UR and cannot be set again until UR is cleared.

Sleep (SLEEP)

The SLEEP bit is used to control the RX features of the multiprocessor modes. This bit is reset to 0 by UR.

Serial Clock Source (CLK)

The CLK determines the source of SCLK. Setting this bit to a 0 selects an external SCLK, which is input on the high impedance A(6) line. Setting it to a 1 selects an internal SCLK, derived from Timer 3. This signal is output on the low impedance A(6) line. The CLK bit is reset to 0 by UR and cannot be set again until UR is cleared.

Timer 3 Data Register

The Timer 3 Data register (T3DATA) is a read/write register and is accessed through P20 in the Peripheral File.

T3DATA

7	6	5	4	3	2	1	0

8-bit timer value (read)

8-bit timer latch (write)

Receiver Buffer

The receiver buffer (RXBUF) is a read-only register and is accessed through P22 in the Peripheral File. This register contains the current data from the RX. Writing has no direct effect on this register. Note that the read/write sequence of the MOVP instruction (as well as ORP, XORP, ANDP) performs a read before a write. This action will perform a spurious clearing of the RXBUF, and will set RXRDY to 0. Data in the RXBUF is right justified with padded 0s.

RXBUF

7	6	5	4	3	2	1	0

0 0 0 ← 5 data bits →
0 0 ← 6 data bits →
0 ← 7 data bits →
← 8 data bits →

Transmitter Buffer

The transmitter buffer (TXBUF) is a write-only register and is accessed through P23 in the Peripheral File. This register contains the data to be transmitted by the TX. Reading P23 returns >00. Data written to the TXBUF must be right-justified since the leftmost bits will be ignored for characters less than eight bits in length.

TXBUF

7	6	5	4	3	2	1	0

x x x ← 5 data bits →
x x ← 6 data bits →
x ← 7 data bits →
← 8 data bits →

Serial Port Initialization

To use the serial port on the 7001/7041 the user must first initialize it. After initialization, the serial port is operated by simply reading and writing to Peripheral File registers. Initialize the serial port as follows:

a. Set B3 Data value to 1. This allows the TXD line to transmit.
b. Write to the SMODE register (P17). This sets the character format and the type of communication mode.
c. Write to the SCILO register (second write to P17) to set the UR bit to 0. This same write can also enable the transmitter, receiver, or both.

Once the serial port is initialized it can be operated continuously in the selected operational mode. If the mode needs to be changed, the serial port must be reset, and then reinitialized for the desired mode. The serial port can be reset in two ways; hardware reset (via RESET pin), or software reset (via UR bit in SCILO).

Serial Port Interrupt

INT4 is dedicated to the serial port. Three sources can generate an interrupt through INT4; the transmitter (TX), the receiver (RX), and Timer 3 (T3). Setting TXEN to a 1 allows data loaded into the TXBUF to be shiftd into the TXSHF. The TX sets TXRDY and INT4 FLG to 1 when TXSHF is loaded from TXBUF.

In the communication modes, if RXEN is set to 1 the RX sets RXRDY and INT4 FLG to a 1 when RXBUF is loaded from RXSHF. If RXEN is 0, the RXSHF still receives frames and shifts them into the RXBUF, but RXRDY and INT4 FLG are held to 0. If a character is in RXBUF, and RXEN is then set to a 1, RXRDY and INT4 FLG will be set to 1.

In the serial I/O mode the RXEN is set to initiate the reception of a frame. When the last bit of the frame is received RXEN is reset to 0. However RXRDY and INT4 FLG are still set to 1 when the character is shifted from RXSHF to RXBUF. RXRDY and INT4 FLG bits are not masked by RXEN.

Timer 3, when it prescaler and timer decrement through 0 together sets T3FLG, and also INT4 FLG if T3ENB = 1.

When INT4 is acknowledged by the CPU, RXRDY, TXRDY, and T3FLG are the flags to indicate its source. The INT4 service routine must determine which of these sources caused INT4 in the specific application. For example, if all three are likely sources, the INT4 service routine must check for the following possible situation:

a. RXRDY only
b. TXRDY only
c. T3 only
d. RXRDY, TXRDY, T3
e. RXRDY, TXRDY
f. RXRDY, T3
g. TXRDY, T3
h. None

ELECTRICAL CHARACTERISTICS

MAXIMUM RATINGS*

Supply Voltage, V_{CC}..... -0.3 to +7V
All Input Voltages With Respect To V_{SS}..... -0.3 to +20V
All Output Volgtages With Respect To V_{SS}..... -0.3 to +7V
Power Dissipation..... 1W
Ambient Temperature Under Bias..... -0°C to +70°C
Storage Temperature Range..... -55°C to +150°C

Standard Conditions (unless otherwise stated):
Operating Temperature T_A = 0°C to +70°C

*Exceeding these ratings could cause permanent damage to the device. This is a stress rating only and functional operation of this device at these conditions is not implied - operating ranges are specified in Standard Conditions. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Data labeled "typical" is presented for design guidance only and is not guaranteed.

DC CHARACTERISTICS

Characteristics	Sym	Min	Typ	Max	Units	Conditions
Supply Voltage	V_{DD}	4.5	5	5.5	V	
Supply Current	I_{DD}	-	80	150	mA	All outputs open
High-Level Input Voltage CLOCKIN	V_{IH}	2.6	-	-	V	
High-Level Input Voltage except CLOCKIN	V_{IH}	2.0	-	-	V	
Low-Level Input Voltage CLOCKIN	V_{IL}	-	-	0.6	V	
Low-Level Input Voltage except CLOCKIN	V_{Il}	-	-	0.8	V	
High-Level Output Current	I_{OH}	-	-	-400	µA	
Low-Level Output Current	I_{OL}	-	-	10	mA	
Input Current, Pin A5, MC, $\overline{RST}$, $\overline{INT3}$, $\overline{INT1}$, XTAL2/CLKIN	I_I	-	±2	±10	µA	V_I = V_{SS} to V_{DD}
Input Current, Ports C and D, A0-A4, A6, A7	I_I	-	±10	±100	µA	V_I = 0.4V to V_{DD}
Input Capacitance	C_I	-	2	-	pf	
High-Level Output Voltage	V_{OH}	2.4	2.8	-	V	I_O = -400µA
Low-Level Output Voltage	V_{OL}	-	0.2	0.4	V	I_O = 3.2mA

AC CHARACTERISTICS

Characteristics	Sym	Min	Typ	Max	Units	Conditions
CRYSTAL/CLOCKIN Frequency (divide-by-4-option)	f_{osc}	2.0	-	10.1	MHz	
CRYSTAL Frequency (divide-by-2 option)	f_{osc}	1.0	-	5.05	MHz	Note 1
CLOCK Cycle Time (divide-by-4 option)	$t_c(P)$	99	-	500	ns	
CLOCK Cycle Time (divide-by-2 option)	$t_c(P)$	198	-	1000	ns	
Internal State Cycle Time	$t_c(S)$	396	-	2000	ns	
CLOCKIN Pulse Width High	$t_w(PH)$	45	-	-	ns	
CLOCKIN Pulse Width Low	$t_w(PL)$	45	-	-	ns	
CLOCKIN Rise Time	t_r	-	-	30	ns	Note 2
CLOCKIN Fall Time	t_f	-	-	30	ns	Note 2
CLOCKIN Rise to CLOCKOUT Rise Delay	$t_d(PH-CL)$	-	125	200	ns	
Output Rise Time	$t_r(O)$	-	9	50	ns	
Output Fall Time	$t_f(O)$	-	10	60	ns	

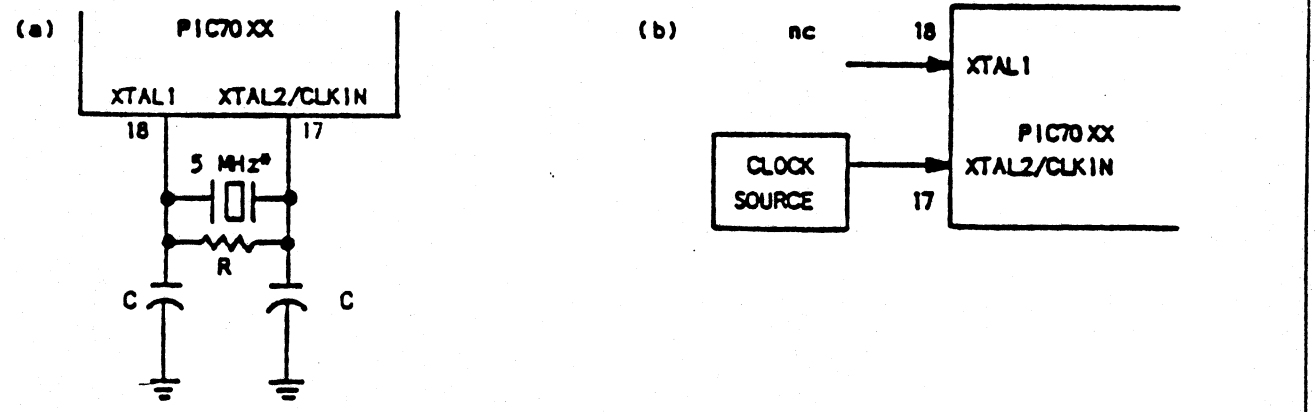
NOTES:
1. Divide-by-4 option recomended with external clock drive.
2. Rise and fall times are measured between the maximum low level and the minimum high level using the 10% and 90% points. Outputs have 100pf loads to V_{SS}.

Memory Interface Timing At 10 MHz Over Full Operating Free-Air Temperature Range

Characteristic	Sym	Min	Typ	Max	Units	Conditions
CLOCKOUT Cycle Time	$t_c(C)$	400	-	2000	ns	Note 1
CLOCKOUT High Pulse Width	$t_w(CH)$	130	170	200	ns	
CLOCKOUT Low Pulse Width	$t_w(CL)$	150	190	240	ns	
CLOCKOUT Rising To ALATCH Falling Edge	$t_d(CH-JL)$	260	300	340	ns	
CLOCKOUT Rising To $\overline{ENA}$ Falling	$t_d(CH-EL)$	-10	15	50	ns	
ALATCH High Pulse Width	$t_w(JH)$	150	190	230	ns	
High Address Valid Before ALATCH fall	$t_d(AH-JL)$	50	170	220	ns	
Low Address Valid Before ALATCH Fall	$t_d(AL-JL)$	50	150	220	ns	
Low Address Hold After ALATCH Fall	$t_h(JL-AL)$	30	45	80	ns	
RD/$\overline{WR}$ Valid Before ALATCH Fall	$t_d(RW-JL)$	50	140	200	ns	
RD/WR Hold After $\overline{ENA}$ Rise	$t_h(EH-RW)$	40	100	-	ns	
High Address Hold After $\overline{ENA}$ Rise	$t_h(EH-AH)$	30	40	-	ns	
Data Out Hold After $\overline{ENA}$ Rise	$t_h(EH-Q)$	65	80	-	ns	
Data Out Valid Before $\overline{ENA}$ Rise	$t_d(Q-EH)$	230	290	-	ns	
$\overline{ENA}$ Fall After Low Address HI-Z	$t_d(AF-EL)$	0	30	120	ns	
$\overline{ENA}$ Rising To Next Address Drive	$t_d(EH-AF)$	60	85	-	ns	
Data In After $\overline{ENA}$ Falling	$t_d(EL-D)$	155	190	-	ns	
Data In Hold After $\overline{ENA}$ Rise	$t_h(EH-D)$	0	-	-	ns	
Access Time, Data In From Valid Address	$t_d(A-D)$	400	470	-	ns	
ENABLE High After Address Valid $\overline{ENA}$	$t_d(A-EH)$	580	-	730	ns	

Note 1: $t_c(C)$ is defined to be $4/f_{osc}$ (or $2/f_{osc}$ if the divide-by-2 option is selected) and may be referred to as a machine state, or simply a state.

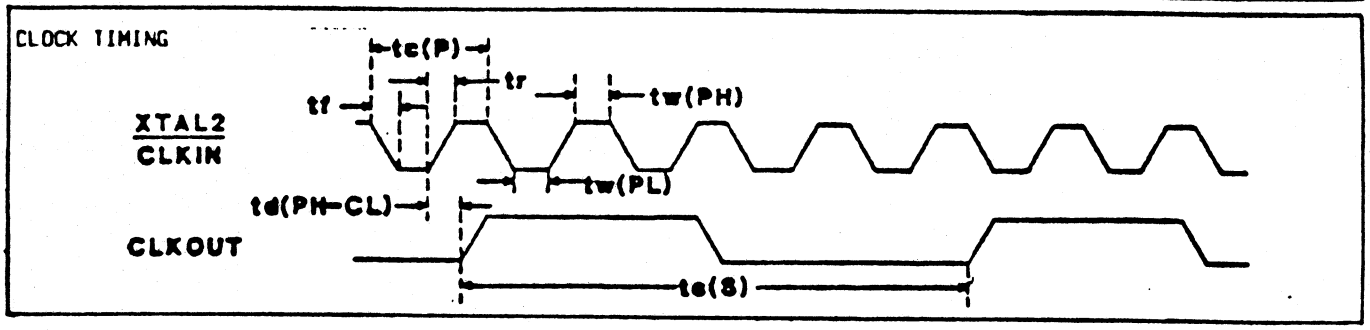
RECOMMENDED CLOCK CONNECTIONS



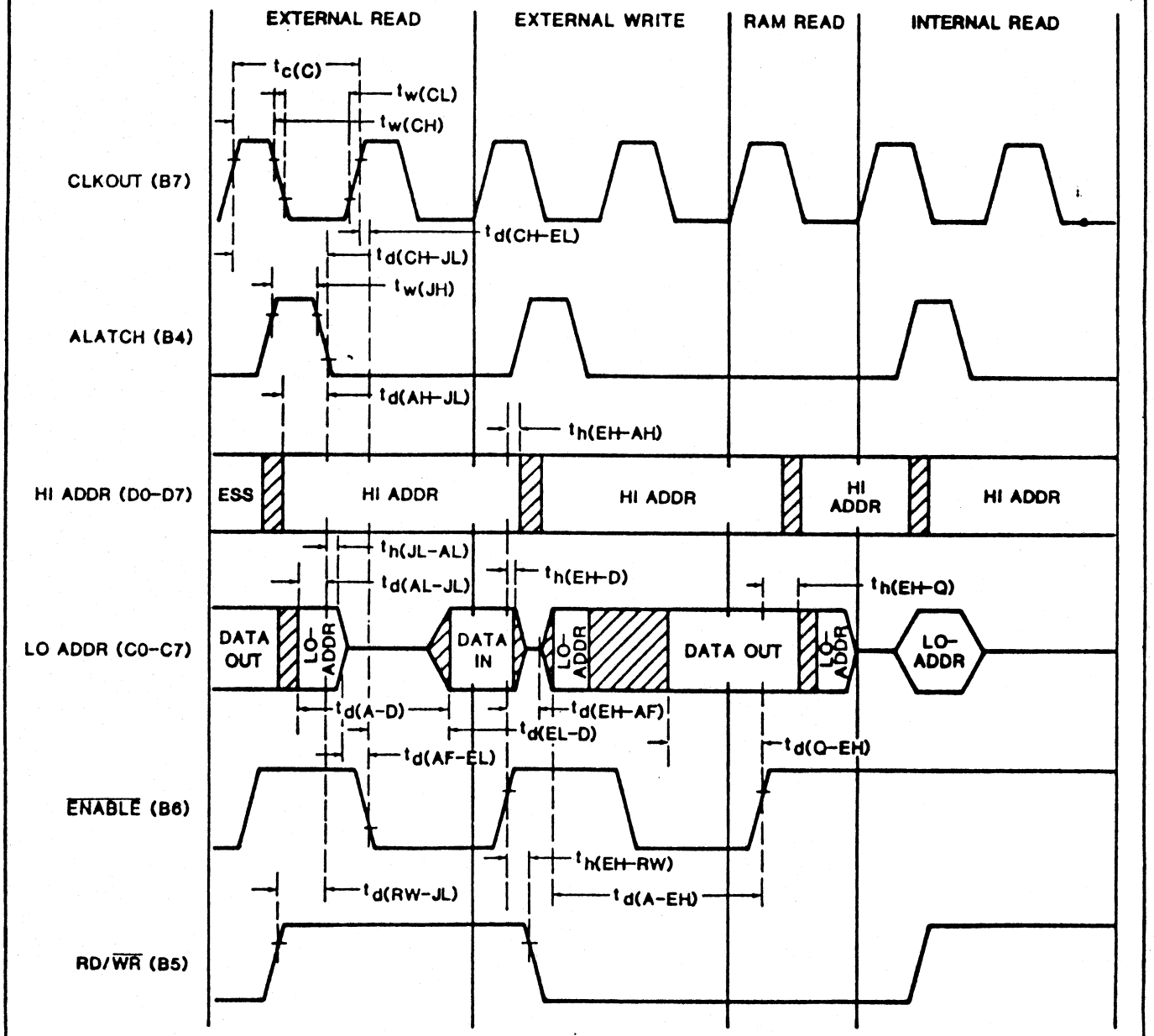
Note: The divide-by-2 input can be used with XTAL only. Divide-by-4 can be used with XTAL or CLKIN inputs.

*Crystal or ceramic resonator, parallel resonant. For XTAL, no resistor used, C = 15pF. For ceramic resonator, R=1M , C=30-68pF.

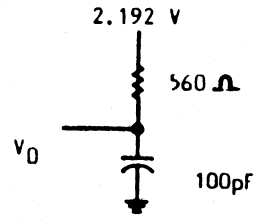
CLOCK TIMING



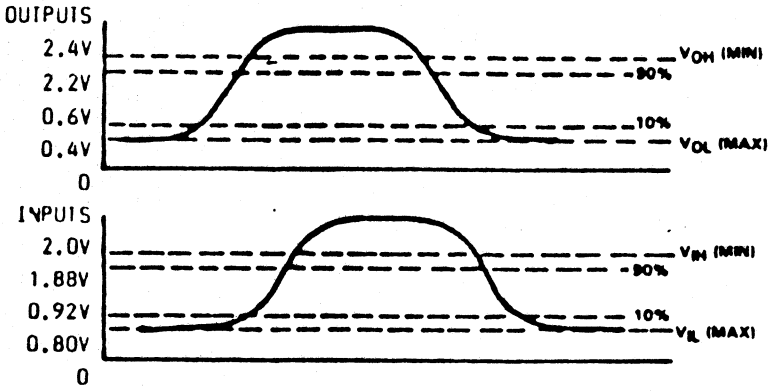
read and write cycle timing



OUTPUT LOADING CIRCUIT FOR TEST

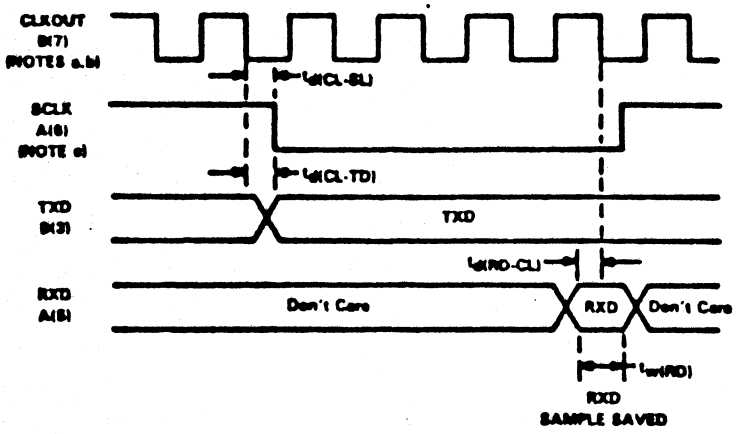


MEASUREMENT POINTS FOR SWITCHING CHARACTERISTICS



SERIAL PORT TIMING

Internal Serial Clock



NOTES:

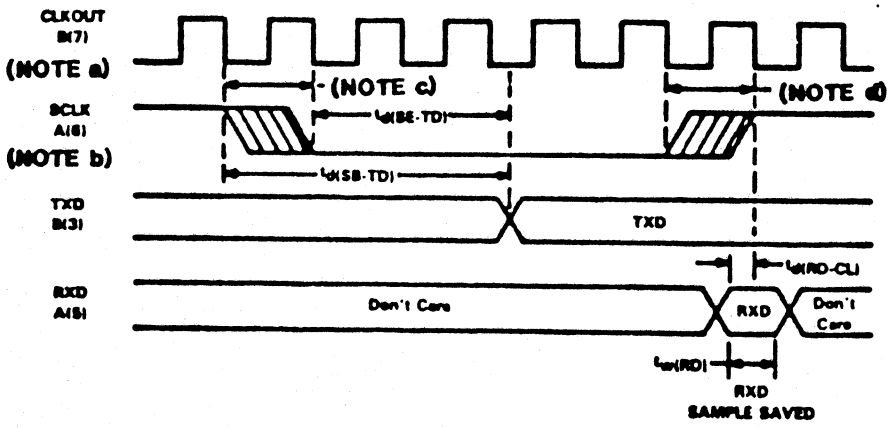
a) The CLKOUT signal is not available in Single-Chip mode.

b) CLKOUT = $t_{c(C)}$ = Ø

c) Example shows SCLK = Ø/8.

Characteristics	Sym	Min	Typ	Max	Units	Conditions
CLKOUT low to SCLK low	$t_{d(Cl-SL)}$	-	$1/4t_{c(C)}$	-	ns	
CLKOUT low to new TXD data	$t_{d(Cl-TD)}$	-	$1/4t_{c(C)}$	-	ns	
RXD data valid before CLKOUT low	$t_{d(RD-Cl)}$	-	$1/4t_{c(C)}$	-	ns	
RXD data valid time	$t_{w(RD)}$	-	$1/2t_{c(C)}$	-	ns	

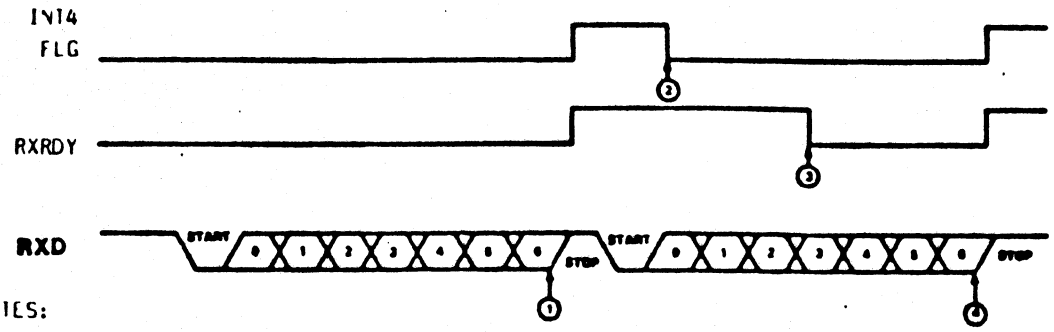
External Serial Clock



NOTES:
a) The CLKOUT signal is not available in Single-Chip mode. CLKOUT = $t_{c(C)} = \emptyset$
b) Example shows SCLK = $\emptyset/10$.
c) SCLK sampled; if 1 then 0, fall transition found.
d) SCLK sampled; if 0 then 1, rise transition found.

Characteristics	Sym	Min	Typ	Max	Units	Conditions
RXD data valid before CLKOUT low	$t_{d(RD-CL)}$	-	$1/4t_{c(C)}$	-	ns	
RXD data valid time	$t_{w(RD)}$	-	$1/2t_{c(C)}$	-	ns	
Start of SCLK sample to new TXD data	$t_{d(SB-TD)}$	-	$3\ 1/4t_{c(C)}$	-	ns	
End of SCLK sample to new TXD data	$t_{d(SL-TD)}$	-	$2\ 1/4t_{c(C)}$	-	ns	

RX Signals In Communication Modes

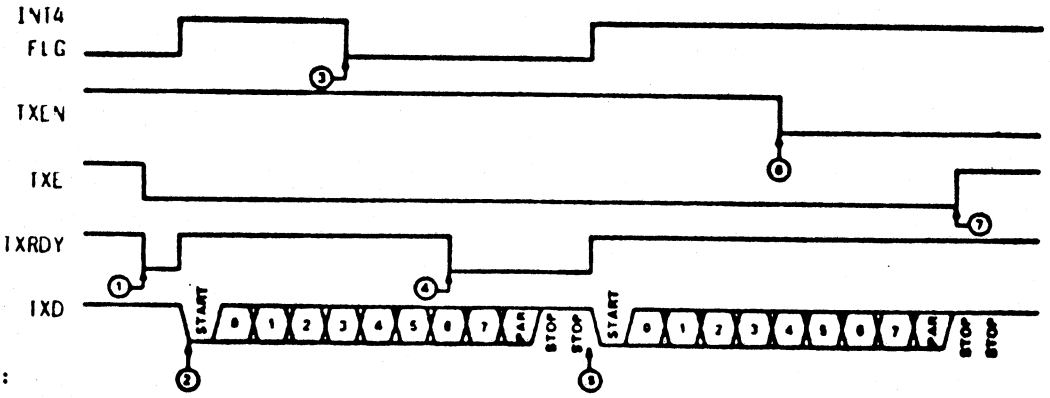


NOTES:
a) Format shown is start bit + seven data bits + stop bits.
b) SCLK is continuous, external or internal.
c) User means user software executed by CPU.
d) If RXEN = 0, RXSHF still receives data from RXD, however, the data is not transferred to RXBUF and RXRDY and INT4 FLG are not set.

SEQUENCE OF EVENTS

1) RXSHF data is transferred to RXBUF. Error status bits are set if an error is detected.
2) User writes to INT4 CLR to clear INT4 FLG. If not, CPU clears.
3) INT4 FLG on entry to Level 4 interrupt routine.
4) User reads RXBUF.

TX Signal In Communication Modes

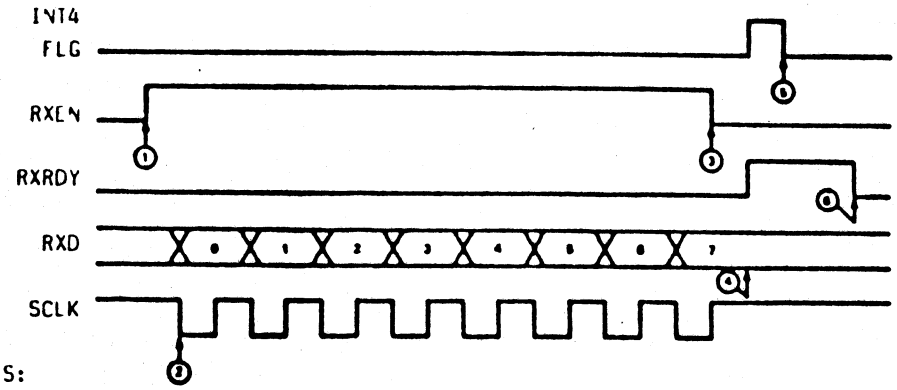


NOTES:
a) Format shown is start plus eight data parity bits two stop bits.
b) SCLK is continuous whether internal or external.
c) User means user software executed by CPU.

SEQUENCE OF EVENTS

1) } User writes to TXBUF.
4) }
2) } TXBUF and WU data is tranferred to TXSHF and WUI and
5) } INT4 FLG and TXIDY are set.
6) User resets TXEN; current frame will finish and transmission will stop whether TXBUF is full or empty.
7) TXE is set if TXBUF and TXSFI are empty.
3) User writes to INT4 CLR to clear INT4 FLG or CPU clears INT4 FLG on entry to level 4 interrupt routine.

RX Signals in Serial I/O Mode

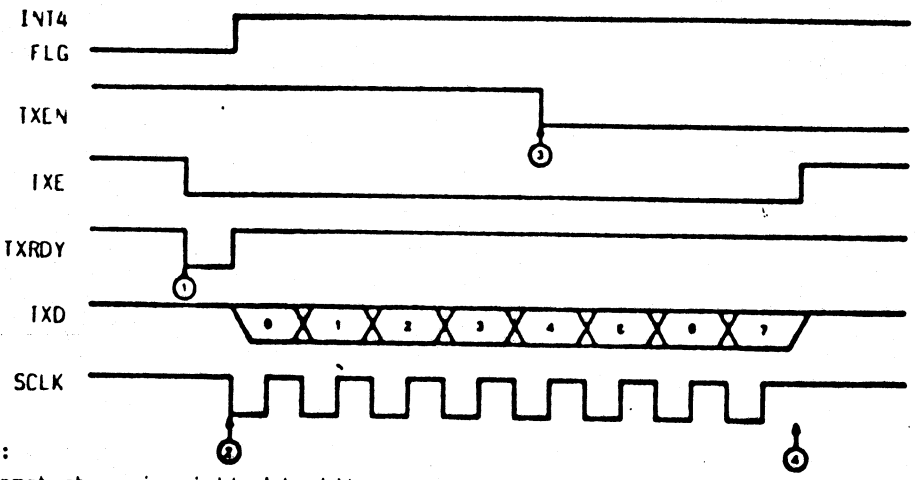


NOTES:
a) RXEN has no effect on INT4 FLG or RXRDY in serial I/O mode.
b) RXD is sampled on CLSK rise; external shift registers should be clocked on SCLK fall.
c) The SCLK source should be internal as it is gated by internal circuitry.

SEQUENCE OF EVENTS

1) User starts receiving by setting RXEN.
2) Gated SCLK starts and data is received.
3) RXEN is automatically cleared in last data bit.
4) RXSHF data is transferred to RXBUF and RXRDY and INT4 are set.
5) User writes INT4 CLR to clear INT4 FLG; if not CPU clears INT4 FLG on entry to level 4 interrupt routine.
6) User reads RXBUF.

TX Signals In Serial I/O Mode



NOTES:
a) Format shown is eight data bits.
b) The SCLK source should be internal as it is gated by internal circuitry.

SEQUENCE OF EVENTS

1) User writes to TXBUF.
2) TXBUF data is transferred to RXSFT; INT4 FLG and RXRDY are set and SCLK starts.
3) User resets TXEN, current frame will finish and transmission will halt whether TXBUF is full or empty.
4) Frame ends and SCLK stops because TXEN = 0.

*This publication contains information from the "TMS7000 Family Data Manual" ©Texas Instruments Incorporated 1983.